1-1-2014

Optimistic fair exchange in the enhanced chosen-key model

Yang Wang
University of Wollongong, ywang@uow.edu.au

Man Ho Au
University of Wollongong, aau@uow.edu.au

Willy Susilo
University of Wollongong, wsusilo@uow.edu.au

Optimistic fair exchange in the enhanced chosen-key model

Abstract

Optimistic fair exchange (OFE) is a kind of protocol to guarantee fairness for the parties involved in an exchange with the help of an arbitrator. A fundamental work of optimistic fair exchange is to define security models capturing realistic attacks and design schemes secure in practical models. The security models are very essential to ensure that they capture practical situation, which will ensure that the protocols can be adopted in practice. The contributions of this paper are three fold. First, we observe that the existing OFE models do not capture realistic situation, where the adversary can actually observe the full signatures generated by the signer, prior to launching the actual attack. That is to say, the adversary is not provided with the signing oracle, which will produce full signatures generated by the signer. It is commonly believed that the full signatures generated by the signer can be simulated by the full signatures generated by the arbitrator. Unfortunately, we show that this perception is false. Second, we propose an enhanced model of OFE that explicitly provides the adversary with the signing oracle, which outputs the full signatures generated by the signer. We demonstrate the difference between our enhanced model and the existing chosen-key model through two concrete OFE schemes that serve as counterexamples. Finally, we revisit two existing generic constructions of optimistic fair exchange schemes, one based on verifiably encrypted signatures, and the other based on conventional signatures and ring signatures. Our result shows that the two generic approaches can still offer schemes secure in our enhanced model, which captures the real scenario that dishonest users may have access to the full signatures generated by the signer.

Keywords

fair, exchange, enhanced, chosen, key, model, optimistic

Disciplines

Engineering | Science and Technology Studies

Publication Details

Wang, Y., Au, M. Ho. & Susilo, W. (2015). Optimistic fair exchange in the enhanced chosen-key model. Theoretical Computer Science, 562 57-74.

Optimistic Fair Exchange in the Enhanced Chosen-key Model

Yang Wang[a,c], Man Ho Au[b,c,*], Willy Susilo[c,1]

[a] *Cyberspace Security College*
PLA Information Engineering University, China.
[b] *Department of Computing*
Hong Kong Polytechnic University, Hong Kong
[c] *Centre for Computer and Information Security Research*
School of Computer Science and Software Engineering
University of Wollongong, Australia

Abstract

Optimistic fair exchange (OFE) is a kind of protocol to guarantee fairness for the parties involved in an exchange with the help of an arbitrator. A fundamental work of optimistic fair exchange is to define security models capturing realistic attacks and design schemes secure in practical models. The security models are very essential to ensure that they capture practical situation, which will ensure that the protocols can be adopted in practice.

The contributions of this paper are three fold. First, we observe that the existing OFE models do not capture realistic situation, where the adversary can actually observe the full signatures generated by the signer, prior to launching the actual attack. That is to say, the adversary is not provided with the signing oracle, which will produce full signatures generated by the signer. It is commonly believed that the full signatures generated by the signer can be simulated by the full signatures generated by the arbitrator. Unfortunately, we show that this perception is false. Second, we propose an enhanced model of OFE that explicitly provides the adversary with the signing oracle, which outputs the full signatures generated by the signer. We demonstrate the difference between our enhanced model and the

*Corresponding author.
Email addresses: yw990@uowmail.uow.edu.au (Yang Wang), csallen@comp.polyu.edu.hk (Man Ho Au), wsusilo@uow.edu.au (Willy Susilo)

[1]This work is supported by ARC Future Fellowship FT0991397.

existing chosen-key model through two concrete OFE schemes that serve as counterexamples. Finally, we revisit two existing generic constructions of optimistic fair exchange schemes, one based on verifiably encrypted signatures, and the other based on conventional signatures and ring signatures. Our result shows that the two generic approaches can still offer schemes secure in our enhanced model, which captures the real scenario that dishonest users may have access to the full signatures generated by the signer.

Keywords: Optimistic Fair Exchange, Chosen-key Model, Enhanced Model

---

1. Introduction

Nowadays electronic commerce grows rapidly and assumes increasing importance. A significant issue for electronic commerce is how to exchange digital items in a fair way so that either each party receives the other's item or neither party does. Since digital items are normally not revocable, i.e. once the digital item has been sent then there is no means to revoke or cancel it, the exchange of digital items should happen simultaneously to achieve fairness for both involved parties. Unfortunately, real simultaneity in general cannot be realized, since the transmission of any data requires time. Moreover, the networks where the exchange takes place may be insecure and there is no assurance that the digital item will eventually be delivered to the intended recipient.

Optimistic fair exchange (OFE), first introduced by Asokan, Schunter and Waidner [1] in 1997, is a kind of protocol to solve the fair exchange problem with the help of a trusted third party named "an arbitrator". In such a protocol, the arbitrator is used in an effective manner in the sense that it only involves to solve the disputes between participants, while in the vast majority of transactions, the arbitrator does not need to be involved at all. Consider the scenario that Alice is willing to sign some statements, for instance, an electronic check, in exchange for Bob's fulfillment of some obligation (delivers some digital good, for example). By adopting an optimistic fair exchange protocol, this exchange can be summarized as a three-step process. Alice, usually called the signer, firstly sends a partial signature to Bob, usually called the verifier. The partial signature assures Bob that the arbitrator is able to convert it into a full one. Then Bob fulfills his obligation. Later, Alice should send her full signature to complete the exchange. In the case Alice refuses to do so, Bob can ask the arbitrator to make a resolution, who

will convert Alice's partial signature into a full one and send it back to Bob.

In an OFE scheme, the full signatures generated by the signer and those generated by the arbitrator based on the signer's partial signature are both viewed as the signer's valid full signatures and represent the signer's commitment on some statements. However, they does not necessarily to be the same. Following the terms in [2], full signatures generated by the signer are called *actual signatures* and full signatures generated by the arbitrator are called *resolved signatures*.

1.1. Previous Work and Related Notions

Optimistic fair exchange has a long history due to its fundamental role in electronic commerce. It is well-known that optimistic fair exchange schemes can be constructed from *verifiably encrypted signatures* [3, 4, 5, 6, 7, 8] and *sequential two-party multisignatures* [9]. It has been widely accepted that optimistic fair exchange schemes should have the property called "resolution ambiguity" [9, 10, 11], namely the actual signatures generated by the signer should be at least computationally indistinguishable from the resolved signatures generated by the arbitrator. As the intervention of an arbitrator could be caused by a network failure rather than by the cheating of a signer, an optimistic fair exchange scheme with resolution ambiguity property can avoid bad publicity for the signer. In the following, we mainly review the attempts in defining security models capturing possible practical attacks for optimistic fair exchange schemes, as they are mostly relevant to this paper.

Early optimistic fair exchange protocols was studied in the single-user setting and the security model assumed only one signer and one verifier. The first formal security model was proposed in [1, 3] but failed to consider the case where the arbitrator itself may be dishonest. A more generalized model in the single-user setting was suggested by Dodis and Reyzin [9] to take into account a dishonest arbitrator.

Since there are many users who may share the same arbitrator in the real world, the security model in the single-user setting does not capture the possible attacks by colluding dishonest users. In PKC 2007, Dodis, Lee and Yum [10] considered the multi-user security of optimistic fair exchange which allows dishonest users to collude to cheat another user. They separated the security of optimistic fair exchange between single-user setting and multi-user setting by showing that an optimistic fair exchange instance provably secure in the single-user setting is not secure in the multi-user setting. Independently, this was also studied by Zhu, Susilo and Mu in 2007 [12].

Since then, the security of optimistic fair exchange intuitively covers the following three aspects.

- Security against signers: the signer should not be able to generate a partial signature that can not be converted into a full one by the honest arbitrator.
- Security against verifiers: the verifier should not be able to generate a full signature of the signer's by himself/herself.
- Security against the arbitrator: the arbitrator should not be able to generate a full signature on behalf of the signer without observing a corresponding partial one.

Most optimistic fair exchange protocols have been studied in the *certified-key* model (also known as the *registered-key* model [13]). In this model, it is assumed that the authenticity of public keys are verifiable and each user in the system should show its knowledge of the corresponding secret key in the public key registration stage to resist key substitution attacks. That is to say, in this model, the dishonest signer and the dishonest arbitrator have to show their knowledge of their corresponding secret keys, and the dishonest verifier can only make resolution queries with respect to the target signer and other public keys whose secret keys are known.

However, in the public key infrastructure, when a certification authority issues a certificate of a user's public key, the user is not required to show its knowledge of the secret key. Thus the certified-key model is not practical enough, as it relies on a stronger assumption than the normal authenticity assumption placed on the certification authority.

In CT-RSA 2008, Huang, Yang, Wong and Susilo [11] studied the security of optimistic fair exchange in the *chosen-key model*, in which the adversary can adversarially select public keys without knowing the corresponding secret key. This model provides more realistic power to the adversary in attacking the honest users. On the one hand, the adversary can choose its own public key without knowing the corresponding secret key. Specifically, in the security against the arbitrator, the dishonest arbitrator is even allowed to set its own public key after knowing the target signer's public key. In the certified-key model, however, the dishonest arbitrator has to set its key pair before seeing the target signer's public key. On the other hand, for a dishonest signer or verifier, the adversary is allowed to make resolution queries

with respect to arbitrary public keys, without knowing the corresponding secret keys. They demonstrated, through a counterexample, that a provably secure fair exchange in the certified-key model may not be secure in the chosen-key model. A generic optimistic fair exchange construction secure in the chosen-key model based on conventional signatures and ring signatures was also proposed.

1.2. Motivation: the need for the signing oracle in the security model

The most fundamental work of OFE is to define security models, that capture realistic attacks. Based on these models, secure schemes will be subsequently designed. These security models are very essential to ensure that they capture practical situation, which will ensure that the protocols can be adopted in practice. We observe that the existing OFE models in fact do not capture realistic situation, where the adversary (i.e. the dishonest verifiers or the dishonest arbitrator) can actually observe the full signatures generated by the signer before launching the actual attack. This realistic situation implies that in the security model, the adversary should have been provided with the signing oracle, which will produce full signatures generated by the signer.

In all the proposed optimistic fair exchange models, in the security against the arbitrator, the dishonest arbitrator is only given a partial signing oracle, which takes as input a message and outputs the target signer's partial signature on this message. Furthermore, in the security against the verifier, the dishonest verifier is given the partial signing oracle and a resolution oracle, which takes as input a partial signature and outputs a resolved signature. We note that the two aspects of the OFE security have the common feature that the adversary (the dishonest verifiers or the dishonest arbitrator) is not provided the access to the actual signatures. This situation in fact separates the security models of OFE schemes from the realistic situation, which may result in the insecurity of the scheme in practice, which will hinder the adoption of OFE in practice.

To date, it is commonly believed that there is no need to provide the dishonest arbitrator with the signing oracle, as the dishonest arbitrator can generate a full signature by first gaining a partial signature from the partial signing oracle and then converting it into a full one using its own secret key. Similarly, in the models of security against the verifier, the dishonest verifier is given the partial signing oracle and a resolution oracle, which takes as input a partial signature and outputs a full signature, as it is commonly

acknowledged that the signer can gain a full signature by first requesting a partial signature from the partial signing oracle and then asking the resolution oracle to convert it into a full one. Unfortunately, by carefully analyzing the existing models, we notice that the above models at most allow the adversary (the dishonest arbitrator or the dishonest verifier) to gain the partial signatures and the resolved signatures generated by the arbitrator. For a malicious arbitrator who generates its public key based on a target signer and may have no knowledge of its own secret key, it may not even be able to get a resolved signature. Hence, the real scenario that an adversary may also have access to the actual signatures generated by the signer is not captured. The crux of the issue in the security models is the lack of signing oracle access, which has been believed to be unnecessary since the partial signing oracle and the resolution oracle have been provided.

Furthermore, we observe that the notion of resolution ambiguity does not imply that signing oracle is helpless to the adversary. The distinguisher in the definition of resolution ambiguity is not equipped with any power, i.e. it is not offered with any oracle access, not to mention the arbitrator's secret key. In other words, the resolution ambiguity property only guarantees a preliminary level of computational indistinguishability between the actual signature and the resolved signature. There is no guarantee that the actual signature is still computationally indistinguishable from the resolved signature in the view of the dishonest verifier, who can have partial signing oracle access and resolution oracle access, let alone in the view of the dishonest arbitrator, who can have partial signing oracle access and choose the arbitration key pair himself.

1.3. Our Contributions

In this paper we propose a new security model for optimistic fair exchange that for the first time provides an adversary with the signing oracle, which takes as input a message and outputs the actual signatures. Compared with the existing models, our model explicitly captures the real scenario that an adversary may have access to the full signatures generated by the signer. We call our new security model as an *enhanced chosen-key model*, which can be viewed as an enhancement of the model of [11].

Next, to show the necessity of our enhanced model, in Section 5, we show that, even for a honest but curious arbitrator (the arbitrator correctly generates its key pair and knows its own secret arbitration key), the security in the multi-user setting and chosen-key model does not imply that in our enhanced

model with a concrete optimistic fair exchange scheme that serves as a counterexample. In other words, the current definition of resolution ambiguity (the actual signatures generated by the signer should be computationally indistinguishable from the resolved signatures generated by the arbitrator) fails to guarantee that deprivation of the signing oracle is reasonable.

Since the current definition of resolution ambiguity fails to justify the absence of the signing oracle, a natural and straightforward question would be "if we further strengthen the definition of resolution ambiguity, would it be possible to fill the gap between our enhanced model and existing model?" Unfortunately, the answer is no. Specifically, in Section 6, we demonstrate that, for a malicious arbitrator (the arbitrator adversarially chooses its arbitration public key and without knowing the corresponding secret key), a provably secure fair exchange in the existing model is not secure in our enhanced model, even if the definition of resolution ambiguity is strengthened to the highest level, i.e., the actual signatures generated by the signer and the resolved signatures generated by the arbitrator have the identical distribution. This answer about the question firmly certifies the significance of our proposal of the enhanced chosen-key model.

Since our enhanced model is more realistic and provides the adversary with an extra power of querying the signing oracle, a natural and fundamental question would be whether or not a scheme secure in our enhanced model exists. We provide an affirmative answer to this question in Section 7 by showing that fortunately two well-known generic construction of optimistic fair exchange (one is based on verifiably encrypted signatures, and the other is based on conventional signatures and ring signatures) can still result in schemes secure in our enhanced model.

2. Cryptographic Notions and Assumptions

In this section, we review several well known notions and computational assumptions that will be used throughout the paper.

Definition 1. (Negligible function). *A function $\mu : \mathbb{N} \to \mathbb{R}$ is negligible if for any positive integer c there exists an integer N_c such that for all $k > N_c$,*

$$\mu(k) < \frac{1}{k^c}.$$

Definition 2. (Computational Indistinguishability). *Two distribution families* $\{X_k\}_{k\in\mathbb{N}}$ *and* $\{Y_k\}_{k\in\mathbb{N}}$ *are computationally indistinguishable if for every probabilistic polynomial time algorithm* $\mathcal{A}$*, the following quantity*

$$\mu(k) = \big| \Pr_{x\in X_k}[\mathcal{A}(x) = 1] - \Pr_{x\in Y_k}[\mathcal{A}(x) = 1] \big|$$

is a negligible function in k*.*

2.1. Bilinear pairing

Let G, G_T be two cyclic groups of the same order p for some large prime p. We say that e is a bilinear map [14] if $e : G \times G \to G_T$ satisfies the following properties.

- (Bilinearity) For all elements of $g, h \in G$, $a, b \in \mathbb{Z}_p$, it holds that

$$e(g^a, h^b) = e(g, h)^{ab}.$$

- (Non-degeneracy) There exists $g, h \in G$ such that $e(g, h)$ is not the identity element of G_T.

- (Efficiency) The group operation in G and the map e are both efficiently computable.

2.2. Complexity Assumptions

Let G be of prime order $p \geq 2^k$, where k is a security parameter. Let g be a generator of G, and u be a random element of G. Let further x be a random element in $\mathbb{Z}_p$ and $\mathbf{Q} =< g, g^x, u, \{(g^{1/(x+s_i)}, g^{s_i}, u^{s_i})\}_{i=1}^{q} >$ where s_i is uniformly randomly chosen from $\mathbb{Z}_p$ for $1 \leq i \leq q$. Below we review three computational assumptions that will be throughout this paper.

Definition 3. (q-SDH Assumption [15]). *The q-Strong Diffie-Hellman (q-SDH) assumption holds for* G *if for every probabilistic polynomial time adversary* $\mathcal{A}$ *with input* $(g, g^x, g^{x^2}, \cdots, g^{x^q})$*, the advantage of* $\mathcal{A}$ *in computing a tuple* $(s, g^{\frac{1}{x+s}})$

$$\mathsf{Adv}_{\mathcal{A}}(k) = \Pr[\mathcal{A}(g, g^x, g^{x^2}, \cdots, g^{x^q}) = (s, g^{\frac{1}{x+s}})]$$

is negligible in k*.*

Definition 4. (q-HSDH Assumption [16]). *The q-Hidden Strong Diffie-Hellman (q-HSDH) assumption holds for G if for every probabilistic polynomial time adversary $\mathcal{A}$ with input $\boldsymbol{Q}$, the advantage of $\mathcal{A}$ in computing another triple $(g^{1/(x+s)}, g^s, u^s)$*

$$\mathsf{Adv}_{\mathcal{A}}(k) = \Pr[\mathcal{A}(\boldsymbol{Q}) = (g^{1/(x+s)}, g^s, u^s)]$$

is negligible in k, where $s \in \mathbb{Z}_p$ and $s \notin \{s_1, \cdots, s_q\}$.

Definition 5. (q-DHSDH Assumption [17]). *The q-Decisional Hidden Strong Diffie-Hellman (q-DHSDH) assumption holds for G if for every probabilistic polynomial time adversary $\mathcal{A}$, the advantage of $\mathcal{A}$ in distinguishing $(\boldsymbol{Q}, u^s, g^{1/(x+s)})$ from $(\boldsymbol{Q}, u^s, Z)$*

$$\mathsf{Adv}_{\mathcal{A}}(k) = \left| \Pr[\mathcal{A}(\boldsymbol{Q}, u^s, g^{1/(x+s)}) = 1] - \Pr[\mathcal{A}(\boldsymbol{Q}, u^s, Z) = 1] \right|$$

is negligible in k, where s is a random element in $\mathbb{Z}_p$ and Z is a random element of G.

3. Definition of Optimistic Fair Exchange

In this section, we review the definition for a non-interactive optimistic fair exchange (OFE) in the multi-user setting and chosen-key model [11].

Definition 6. *A non-interactive optimistic fair exchange scheme involves users (signers and verifiers) and the arbitrator, and consists of the following (probabilistic) polynomial-time algorithms:*

- $\mathsf{Setup}^{\mathsf{TTP}}$*: This algorithm is run by the arbitrator to generate its secret key* ASK *and public key* APK *on input security parameter* 1^k*.*
- $\mathsf{Setup}^{\mathsf{User}}$*: This algorithm is run by a user to generate its secret key* SK *and public key* PK*. For a user U_i, we use* $(\mathsf{SK}_i, \mathsf{PK}_i)$ *to denote its key pair. The input of this algorithm is the security parameter* 1^k *and (optionally)* APK*.*
- Sig *and* Ver*: These two algorithms are used to generate and verify a full signature respectively.* $\mathsf{Sig}(m, \mathsf{SK}_i, \mathsf{APK})$ *was run by the signer U_i to output a full signature σ on message m, while* $\mathsf{Ver}(m, \sigma, \mathsf{PK}_i, \mathsf{APK})$*, run by a verifier, outputs $\top$ or $\perp$, indicating σ is a valid full signature generated by the signer U_i on m or not.*

- PSig *and* PVer*: These two algorithms are used to generate and verify a partial signature respectively.* $\mathsf{PSig}(m, \mathsf{SK}_i, \mathsf{APK})$ *was run by the signer* U_i *to output a partial signature* σ_P *on message* m*, while* $\mathsf{PVer}(m, \sigma_P, \mathsf{PK}_i, \mathsf{APK})$*, run by a verifier, outputs* $\top$ *or* $\perp$*.*

- Res*: This algorithm is run by the arbitrator to make a resolution.* $\mathsf{Res}(m, \sigma_P, \mathsf{ASK}, \mathsf{PK}_i)$ *outputs either a full signature* σ *on message* m *with respect to the partial signature, or* $\perp$ *indicating failure.*

Correctness of an OFE scheme means that the verification algorithms function properly if the inputs are generated honestly. More formally, it means that

$$\begin{aligned} \mathsf{Ver}(m, \mathsf{Sig}(m, \mathsf{SK}_i, \mathsf{APK}), \mathsf{PK}_i, \mathsf{APK}) &= \top, \\ \mathsf{PVer}(m, \mathsf{PSig}(m, \mathsf{SK}_i, \mathsf{APK}), \mathsf{PK}_i, \mathsf{APK}) &= \top, \ \textit{and} \\ \mathsf{Ver}(m, \mathsf{Res}(m, \mathsf{PSig}(m, \mathsf{SK}_i, \mathsf{APK}), \mathsf{ASK}, \mathsf{PK}_i), \mathsf{PK}_i, \mathsf{APK}) &= \top. \end{aligned}$$

Resolution ambiguity property states that any "resolved signature" outputted by the algorithm $\mathsf{Res}(m, \mathsf{PSig}(m, \mathsf{SK}_i, \mathsf{APK}), \mathsf{ASK}, \mathsf{PK}_i)$ *run by the arbitrator is computationally indistinguishable from the "actual signature" outputted by algorithm* $\mathsf{Sig}(m, \mathsf{SK}_i, \mathsf{APK})$ *run by the signer.*

We emphasize that in the definition of resolution ambiguity, the distinguisher does not have any adaptive adversarial capability. Specifically, the distinguisher is neither provided with any oracle access nor the arbitrator's secret key. Thus, the resolution ambiguity property only guarantees a preliminary level of computational indistinguishability between the actual signature and the resolved signature. There is no guarantee that the actual signature is still computationally indistinguishable from the resolved signature in the view of an adversary who can have partial signing oracle access and resolution oracle access, let alone an adversary, who can have partial signing oracle access and the arbitrator's secret key.

3.1. Security in Multi-User setting and Chosen-key Model

The security in the multi-user setting and chosen-key model [11] guarantees the highest level of security for optimistic fair exchange in literatures. Typically, the security consists of three aspects: security against signers, security against verifiers, and security against the arbitrator, which considered the

cases when the signer, the verifier and the arbitrator are dishonest, respectively. We review this security model below.

SECURITY AGAINST SIGNERS. To guarantee that no signer can gain advantage over the verifier, we require the probability that any PPT adversary $\mathcal{A}$ succeeds in the following experiment is negligible in k.

$$\begin{array}{rcl} \mathsf{Setup}^{\mathsf{TTP}}(1^k) & \rightarrow & (\mathsf{ASK}, \mathsf{APK}) \\ (m, \sigma_P, \mathsf{PK}^*) & \leftarrow & \mathcal{A}^{O_{\mathsf{Res}}}(\mathsf{APK}) \\ \sigma & \leftarrow & \mathsf{Res}(m, \sigma_P, \mathsf{ASK}, \mathsf{PK}^*) \\ \text{success of } \mathcal{A} & := & \left[\begin{array}{c} \mathsf{PVer}(m, \sigma_P, \mathsf{PK}^*, \mathsf{APK}) = \top \\ \mathsf{Ver}(m, \sigma, \mathsf{PK}^*, \mathsf{APK}) = \bot \end{array} \right] \end{array}$$

In this experiment, O_{Res} is the resolution oracle, which takes as input a message m, a partial signature σ_P and a signer's public key PK_i and returns the output of the algorithm $\mathsf{Res}(m, \sigma_P, \mathsf{ASK}, \mathsf{PK}_i)$. In the resolution oracle queries, the adversary can make queries with respect to any adversarially chosen public key, without requiring to know the corresponding secret key. This aspect of security guarantees that no signer should be able to produce a partial signature that can be verified as valid but cannot be resolved into a valid full one by the honest arbitrator.

SECURITY AGAINST VERIFIERS. To guarantee that no verifier can gain advantage over the signer, we require the probability that any PPT adversary $\mathcal{A}$ succeeds in the following experiment is negligible in k.

$$\begin{array}{rcl} \mathsf{Setup}^{\mathsf{TTP}}(1^k) & \rightarrow & (\mathsf{ASK}, \mathsf{APK}) \\ \mathsf{Setup}^{\mathsf{User}}(1^k) & \rightarrow & (\mathsf{SK}, \mathsf{PK}) \\ (m, \sigma) & \leftarrow & \mathcal{A}^{O_{\mathsf{PSig}}, O_{\mathsf{Res}}}(\mathsf{PK}, \mathsf{APK}) \\ \text{success of } \mathcal{A} & := & \left[\begin{array}{c} \mathsf{Ver}(m, \sigma, \mathsf{PK}, \mathsf{APK}) = \top \\ (m, \cdot) \notin Query(\mathcal{A}, O_{\mathsf{Res}}) \end{array} \right] \end{array}$$

In this experiment, $Query(\mathcal{A}, O_{\mathsf{Res}})$ is the set of queries made by $\mathcal{A}$ to the resolution oracle O_{Res}, which is the same as described in the experiment of security against signers. The partial signing oracle O_{PSig} takes as input a message m and outputs a partial signature σ_P generated using the challenge signer's secret key SK. This aspect of security guarantees that no verifier should be able to generate a full signature or convert any partial signature σ_P into a full one by himself. It is widely believed that there is no need

to provide $\mathcal{A}$ with access to the signing oracle O_{Sig}, which takes as input a message m and outputs a full signature σ on m generated using the challenge signer's secret key SK, as it seems that a full signature outputted by O_{Sig} could be simulated by using the oracles O_{PSig} and O_{Res}.

SECURITY AGAINST THE ARBITRATOR. To prevent the arbitrator from issuing full signatures on behalf of the signature arbitrarily, we require that the probability that any PPT adversary $\mathcal{A}$ succeeds in the following experiment is negligible in k.

$$\begin{aligned} \mathsf{Setup}^{\mathsf{User}}(1^k) &\rightarrow (\mathsf{SK}, \mathsf{PK}) \\ (\mathsf{ASK}^*, \mathsf{APK}) &\leftarrow \mathcal{A}(\mathsf{PK}) \\ (m, \sigma) &\leftarrow \mathcal{A}^{O_{\mathsf{PSig}}}(\mathsf{ASK}^*, \mathsf{APK}, \mathsf{PK}) \\ \text{success of } \mathcal{A} &:= \left[\begin{array}{c} \mathsf{Ver}(m, \sigma, \mathsf{PK}, \mathsf{APK}) = \top \\ m \notin \mathit{Query}(\mathcal{A}, O_{\mathsf{PSig}}) \end{array} \right] \end{aligned}$$

In this experiment, ASK^* is $\mathcal{A}$'s state information, which might not be the corresponding secret key of APK. $\mathit{Query}(\mathcal{A}, O_{\mathsf{PSig}})$ is the set of queries made by $\mathcal{A}$ to the partial signing oracle O_{PSig}, which is the same as defined in the previous experiment. This aspect of security guarantees that the arbitrator should not be able to produce a full signature on behalf of the signer on any message, unless the signer has already generated a partial signature on that message. As in the experiment of security against verifiers, the adversary is not given the signing oracle access, as it is believed that the adversary can gain a full signature by firstly receiving a partial signature from the partial signing oracle and then converting it into a full one by using the secret arbitrator key. We emphasize that in the experiment, the adversary can adaptively set APK after it is given the challenge signer's public key PK and it may have no knowledge of the corresponding private key of APK.

4. The Enhanced Chosen-key Model

As discussed in the introduction, the actual signature generated by the signer may be different from the resolved signature generated by the arbitrator based on a corresponding partial one. Thus, it seems natural and straightforward that a security model should capture the real scenario that an adversary may observe the actual signatures generated by the signer. To explicitly take this case into account, we propose the following enhanced

chosen-key model, i.e., in the experiments of security against verifiers and security against the arbitrator, we offer the adversary with the signing oracle access.

SECURITY AGAINST VERIFIERS. We require the probability that any PPT adversary $\mathcal{A}$ succeeds in the following experiment is negligible in k.

$$\begin{array}{rcl} \mathsf{Setup}^{\mathsf{TTP}}(1^k) & \rightarrow & (\mathsf{ASK}, \mathsf{APK}) \\ \mathsf{Setup}^{\mathsf{User}}(1^k) & \rightarrow & (\mathsf{SK}, \mathsf{PK}) \\ (m, \sigma) & \leftarrow & \mathcal{A}^{O_{\mathsf{PSig}}, O_{\mathsf{Sig}}, O_{\mathsf{Res}}}(\mathsf{PK}, \mathsf{APK}) \\ \text{success of } \mathcal{A} & := & \left[\begin{array}{c} \mathsf{Ver}(m, \sigma, \mathsf{PK}, \mathsf{APK}) = \top \\ (m, \cdot) \notin Query(\mathcal{A}, O_{\mathsf{Res}}) \\ m \notin Query(\mathcal{A}, O_{\mathsf{Sig}}) \end{array} \right] \end{array}$$

where O_{PSig}, O_{Res} and $Query(\mathcal{A}, O_{\mathsf{Res}})$ are the same as that in the existing model reviewed in Section 3.1, oracle O_{Sig} takes as input a message m and outputs a full signature σ on m generated using the challenge signer's secret key SK, and $Query(\mathcal{A}, O_{\mathsf{Sig}})$ is the set of queries made by $\mathcal{A}$ to oracle O_{Sig}.

SECURITY AGAINST THE ARBITRATOR. We require that the probability that any PPT adversary $\mathcal{A}$ succeeds in the following experiment is negligible in k.

$$\begin{array}{rcl} \mathsf{Setup}^{\mathsf{User}}(1^k) & \rightarrow & (\mathsf{SK}, \mathsf{PK}) \\ (\mathsf{ASK}^*, \mathsf{APK}) & \leftarrow & \mathcal{A}(\mathsf{PK}) \\ (m, \sigma) & \leftarrow & \mathcal{A}^{O_{\mathsf{PSig}}, O_{\mathsf{Sig}}}(\mathsf{ASK}^*, \mathsf{APK}, \mathsf{PK}) \\ \text{success of } \mathcal{A} & := & \left[\begin{array}{c} \mathsf{Ver}(m, \sigma, \mathsf{PK}, \mathsf{APK}) = \top \\ m \notin Query(\mathcal{A}, O_{\mathsf{PSig}}) \\ m \notin Query(\mathcal{A}, O_{\mathsf{Sig}}) \end{array} \right] \end{array}$$

where ASK^*, O_{PSig} and $Query(\mathcal{A}, O_{\mathsf{PSig}})$ are the same as reviewed in the existing model reviewed in Section 3.1, and O_{Sig} and $Query(\mathcal{A}, O_{\mathsf{Sig}})$ are the same as described in the above experiment.

5. Separation of the Enhanced Model and the Existing Model

In this section, we present a concrete optimistic fair exchange scheme that is secure in the multi-user setting and chosen key model reviewed in Section 3.1 but insecure in our enhanced chosen-key model. This demonstrates

that the common belief that no signing oracle is required to be provided to the adversary is insufficient. That is to say, with this counterexample, we can safely arrive to the conclusion that the existing models fail to capture the real scenario that an adversary may have access to the signers actual signatures, and in the contrary, our enhanced chosen-key model is more complete and practical.

Prior to proposing the concrete optimistic fair exchange scheme that will serve as a counterexample, we will first provide a high level description. The concrete optimistic fair exchange scheme is constructed in the bilinear pairing setting. Let $e : G \times G \to G_T$ be a bilinear pairing where g is a generator of G. The signer's public key is $X = g^x$ and a random $u \in G$. The arbitrator's public key is $Y = g^y$. The signer and the arbitrator keep their secret keys x and y as private, respectively. Let further $H_0 : \{0,1\}^* \to G$ and $H_1 : \{0,1\}^* \to \mathbb{Z}_p$ be two hash functions that will be viewed as random functions. The partial signature on a message m is a conventional signature $(H_0(m)^{1/(x+s)}, g^s, u^s) \in G^3$, which can be seen as a variation of the signatures from [18, 19] in which they employ a programmable hash function [18] to guarantee the security without random oracles.

To fully sign a message m, the signer does the following.

1. The signer firstly generates a partial signature $(H_0(m)^{1/(x+s)}, g^s, u^s)$ on message m.
2. Let $\bar{m}$ be the 'complement' message of m, i.e. $\bar{m}$ and m are of the same bit-length, where for each i-th bit, $\bar{m}_i$ is the complement of m_i.
3. The signer then generates a designated confirmer signature [20, 19] on message $\bar{m}$
$$(\delta', \gamma', \theta') = (H_0(\bar{m})^{1/(x+s')}, Y^{s'}, u^{s'}),$$
with the arbitrator being the confirmer where s' is a random. This confirmer signature can be seen as a variation of the confirmer signature from [19] in which the hash function H_0 is replaced with a programmable hash function.
4. Note that $e(\delta', u)^x = e(H_0(\bar{m}), u)/e(\delta', \theta') := W$. The signer finally makes a non-interactive zero-knowledge proof of knowledge with respect to the message m and the signer's public key (X, u) that it knows the value x such that $g^x = X$ and $e(\delta', u)^x = W$ or it knows the arbitrator's secret key y. The signer can always make such a proof by using its secret key x.

5. The full signature on message m comprises the partial signature on m, the designated confirmer signature on $\bar{m}$ and the non-interactive zero-knowledge proof of knowledge.

To convert a partial signature into a full one, the arbitrator uniformly samples a designated confirmer signature $(\delta', \gamma', \theta') = (Z, Y^{s'}, u^{s'})$ from the signer's designated signature space where $Z \in G$ and an exponent s' are randomly chosen. Let $W = e(H_0(\bar{m}), u)/e(\delta', \theta')$. The arbitrator then makes a non-interactive zero-knowledge proof of knowledge with respect to the message m and the signer's public key (X, u) that it knows the value x such that $g^x = X$ and $e(\delta', u)^x = W$ or it knows the arbitrator's secret key y. The arbitrator can always make the non-interactive zero-knowledge proof of knowledge using its secret key y.

Note that due to the invisibility property of a confirmer signature [20], the designated confirmer signature sampled by the arbitrator is indistinguishable from the one that is generated by the signer. Besides, the non-interactive zero-knowledge proof of knowledge guarantees the proof made by the signer is indistinguishable from the proof made by the arbitrator. Thus the resolution ambiguity property of the concrete optimistic fair exchange scheme holds.

Let G be a multiplicative cyclic group of prime order p, g be a generator of G, and $e : G \times G \to G_T$ be a bilinear pairing where G_T is a multiplicative group of order p. Let $H_0 : \{0,1\}^* \to G$ and $H_1 : \{0,1\}^* \to \mathbb{Z}_p$ be two hash functions. We assume the public parameter (G, p, e, H_1, H_2) is shared by all users. The concrete OFE scheme is as follows.

- $\mathsf{Setup}^{\mathsf{TTP}}(1^k)$: The arbitrator chooses at random $y \in \mathbb{Z}_p$ and computes $Y = g^y$. The public key is set as $\mathsf{APK} = Y$, and the arbitrator keeps $\mathsf{ASK} = y$ as private.

- $\mathsf{Setup}^{\mathsf{User}}(1^k)$: Each user U_i chooses uniformly at random $x_i \in \mathbb{Z}_p$ and $u_i \in G$, and calculates $X_i = g^{x_i}$. The user sets $(\mathsf{SK}_i, \mathsf{PK}_i) = (x_i, (X_i, u_i))$.

- $\mathsf{PSig}(m, \mathsf{SK}_i, \mathsf{APK})$: To partially sign a message m, the user U_i chooses a random $s \in \mathbb{Z}_p$ and returns $\sigma_P := (H_0(m)^{1/(x_i+s)}, g^s, u_i^s)$.

- $\mathsf{PVer}(m, \sigma_P, \mathsf{PK}_i, \mathsf{APK})$: Given a partial signature σ_P from user U_i, the verifier parses σ_P as (δ, v, θ), and returns $\top$ if both $e(v, u_i) = e(g, \theta)$ and $e(\delta, X_i v) = e(H_0(m), g)$ hold. Otherwise, $\perp$ is returned.

- $\mathsf{Sig}(m, \mathsf{SK}_i, \mathsf{APK})$: To fully sign a message m, the user U_i
  1. chooses a random $s \in \mathbb{Z}_p$ and computes $\sigma_P := (H_0(m)^{1/(x_i+s)}, g^s, u_i^s)$.
  2. chooses a random $s' \in \mathbb{Z}_p$ and computes $(\delta', \gamma', \theta') := (H_0(\bar{m})^{1/(x_i+s')}, Y^{s'}, u_i^{s'})$ where $\bar{m}$ is the 'complement' message of m, i.e. for each i-th bit, $\bar{m}_i$ is the complement of m_i.
  3. Denote $W := e(H_0(\bar{m}), u_i)/e(\delta', \theta')$. Note that $W = e(\delta', u_i)^{x_i}$. The user U_i chooses uniformly at random $r, c_1, t_1 \in \mathbb{Z}_p$, and computes $c = H_1(m||\mathsf{PK}_i||g^r||e(\delta', u_i)^r||g^{t_1}Y^{c_1})$, $c_0 = c - c_1 \pmod p$ and $t_0 = r - c_0 x_i \pmod p$.
  4. The full signature is set as $\sigma := (\sigma_P, \delta', \gamma', \theta', c_0, t_0, c_1, t_1)$.

- $\mathsf{Ver}(m, \sigma, \mathsf{PK}_i, \mathsf{APK})$: Given a full signature σ from user U_i, a verifier does as follows.
  1. Check whether $c_0, t_0, c_1, t_1 \in \mathbb{Z}_p$ and $\mathsf{PVer}(m, \sigma_P, \mathsf{PK}_i, \mathsf{APK}) = \top$.
  2. Check whether $e(\gamma', u_i) = e(Y, \theta')$.
  3. Compute $W = e(H_0(\bar{m}), u_i)/e(\delta', \theta')$ and
  $$c = H_1(m||\mathsf{PK}_i||g^{t_0}(X_i)^{c_0}||e(\delta', u_i)^{t_0}W^{c_0}||g^{t_1}Y^{c_1}).$$
  4. Verify whether $c_0 + c_1 = c \pmod p$.
  5. If all the above hold, $\top$ is returned and otherwise, $\perp$ is returned.

- $\mathsf{Res}(m, \sigma_P, \mathsf{ASK}, \mathsf{PK}_i)$: For the user U_i's partial signature σ_P on message m, the arbitrator
  1. first checks whether $\mathsf{PVer}(m, \sigma_P, \mathsf{PK}_i) = \top$. If so, continues; otherwise, returns $\perp$.
  2. chooses at random $s' \in \mathbb{Z}_p$, $Z \in G$ and computes $(\delta', \gamma', \theta') := (Z, Y^{s'}, u_i^{s'})$.
  3. computes $W = e(H_0(\bar{m}), u_i)/e(\delta', \theta')$ where $\bar{m}$ is the complement message of m.
  4. chooses uniformly at random $r, c_0, t_0 \in \mathbb{Z}_p$, and computes
  $$c = H_1(m||\mathsf{PK}_i||g^{t_0}(X_i)^{c_0}||e(\delta', u_i)^{t_0}W^{c_0}||g^r),$$
  $c_1 = c - c_0 \pmod p$ and $t_1 = r - c_1 y \pmod p$.
  5. The full signature is set as $\sigma := (\sigma_P, \delta', \gamma', \theta', c_0, t_0, c_1, t_1)$.

5.1. Security Analysis in the Chosen-key model

It is straightforward to verify that in the above construction, any (partial) signature created by Sig (PSig) will be valid under Ver (PVer), and that any signature created by the arbitrator using Res based on a partial signature generated by PSig will be valid under Ver. Thus, the correctness property of the above construction holds. Based on the q-DHSDH Assumption, the resolution ambiguity property also holds.

Next we show the specific construction is secure in the multi-user setting and chosen-key model reviewed in Section 3.1. Intuitively, the security against signers holds unconditionally as the arbitrator is always able to convert a signer's partial signature into a full one. The security against verifiers holds due to the fact that one cannot make the proof of knowledge without knowing the signer's secret key or the arbitrator's secret key. The security against the arbitrator holds due to the unforgeability of the conventional signature scheme that generates the partial signature.

Theorem 1. *The concrete optimistic fair exchange scheme is unconditionally secure against signers in the multi-user setting and chosen-key model.*

Proof. Obviously, for any message m and any valid signature σ_P on m under the verification key PK_i, the arbitrator can always convert it into a full signature by using its own secret key ASK. Therefore, the security against signers always hold. □

Theorem 2. *The concrete optimistic fair exchange scheme is secure against verifiers in the multi-user setting and chosen-key model if the q-HSDH Assumption holds.*

Proof. Suppose an adversary $\mathcal{A}$ makes q queries to the partial signing oracle and breaks the security against verifiers with non-negligible probability. We show how to construct an algorithm $\mathcal{R}$ that breaks the q-HSDH Assumption.

Note that algorithm $\mathcal{R}$ is given $\mathbf{Q} =< g, g^x, u, \{(g^{1/(x+s_i)}, g^{s_i}, u^{s_i})\}_{i=1}^{q} >$. Its goal is to output another distinct triple $(g^{1/(x+s)}, g^s, u^s)$. Algorithm $\mathcal{R}$ simulates the challenger and interacts with adversary $\mathcal{A}$. Algorithm $\mathcal{R}$ chooses a random integer $x' \in \mathbb{Z}_p$, sets $X' = g^{x'}$, flips a coin and gets a random bit b. According to the random bit b, $\mathcal{R}$ performs one of the following two games.

- Game 0 ($b = 0$): $\mathcal{R}$ forwards $\mathsf{PK} := (g^x, u)$ and $\mathsf{APK} = Y := X'$ to $\mathcal{A}$ and simulates the oracles for $\mathcal{A}$.

  $\mathsf{H_0}$ Queries. At any time adversary $\mathcal{A}$ can query the random oracle H_0. To response to these queries, $\mathcal{R}$ maintains a list of tuples $\langle m_i, a_i \rangle$ as explained below. We refer to this list as H_0-list. The list is initially empty. When $\mathcal{A}$ queries the oracle H_0 on a message $m_i \in \{0,1\}^*$, algorithm $\mathcal{R}$ responds as follows:

  1. If the query m_i already appears on the H_0-list in some tuple $\langle m_i, a_i \rangle$, then algorithm $\mathcal{R}$ responds with $H(m_i) = g^{a_i}$.
  2. Otherwise, $\mathcal{R}$ generates a random $a_i \in \mathbb{Z}_p$, adds the tuple $\langle m_i, a_i \rangle$ to the H_0-list and responds to $\mathcal{A}$ as $H(m_i) = g^{a_i}$.

  $\mathsf{H_1}$ Queries. $\mathcal{A}$ can also query the random oracle H_1. To response to these queries, $\mathcal{R}$ maintains a list of tuples $\langle string^{(i)}, c^{(i)} \rangle$ as explained below. We refer to this list as H_1-list. The list is initially empty. When $\mathcal{A}$ queries the oracle H_1 at a point $string \in \{0,1\}^*$, algorithm $\mathcal{R}$ responds as follows:

  1. If the query $string$ already appears on the H-list in some tuple $\langle string, c \rangle$, then algorithm $\mathcal{R}$ responds with $H(string) = c \in \mathbb{Z}_p$.
  2. Otherwise, $\mathcal{R}$ generates a random $c \in \mathbb{Z}_p$, adds the tuple $\langle string, c \rangle$ to the H_1-list and responds to $\mathcal{A}$ as $H(string) = c \in \mathbb{Z}_p$.

  PSig Queries. For the i-th partial signing query on message m_i, $\mathcal{R}$ checks whether there is a tuple $\langle m_i, a_i \rangle$ in the H_0-list. If not, $\mathcal{R}$ generates a random $a_i \in \mathbb{Z}_p$, adds the tuple $\langle m_i, a_i \rangle$ to the H_0-list. $\mathcal{R}$ returns $((g^{1/(x+s_i)})^{a_i}, g^{s_i}, u^{s_i})$ to $\mathcal{A}$ as the reply for the partial signing query.

  Res Queries. Given a resolution query $(m, \sigma_P, \mathsf{PK}_i)$ where $\mathsf{PK}_i = (X_i, u_i)$ is the signer's public key, algorithm $\mathcal{R}$ responds to this query as follows:

  1. checks whether $\mathsf{PVer}(\sigma_P, m, \mathsf{PK}_i) = \top$. If so, continues; otherwise, $\mathcal{R}$ responds to $\mathcal{A}$ with a special symbol $\perp$.
  2. chooses at random $s' \in \mathbb{Z}_p$, $Z \in G$ and computes $(\delta', \gamma', \theta') := (Z, Y^{s'}, u_i^{s'})$ where $\bar{m}$ is the complement message of m.
  3. computes $W = e(H_1(\bar{m}), u_i)/e(\delta', \theta')$.

4. chooses uniformly at random $c_0, t_0, c_1, t_1 \in \mathbb{Z}_p$, and adds $< m||\mathsf{PK}_i||g^{t_0}(X_i)^{c_0}||e(\delta', u_i)^{t_0}W^{c_0}\ ||g^{t_1}Y^{c_1}, c_0+c_1 >$ to the H_1-list. Since c_0, t_0, c_1, t_1 are all randomly chosen, the probability that $\mathcal{A}$ has previously asked a H_1 query on the string $m||\mathsf{PK}_i||g^{t_0}(X_i)^{c_0}||$ $e(\delta', u_i)^{t_0}W^{c_0}||g^{t_1}Y^{c_1}$ is negligible.
5. The full signature is set as $\sigma := (\sigma_P, \delta', \gamma', \theta', c_0, t_0, c_1, t_1)$ and returned to $\mathcal{A}$ as the reply of the resolution query.

- Game 1 ($b = 1$): $\mathcal{R}$ forwards $\mathsf{PK} := (X', u)$ and $\mathsf{APK} = Y := g^x$ to $\mathcal{A}$ and simulates the oracles for $\mathcal{A}$.

  The H_0, H_1 and resolution queries are simulated the same as in Game 0. For the i-th partial signing query on message m_i, $\mathcal{R}$ checks whether there is a tuple $\langle m_i, a_i \rangle$ in the H_0-list. If not, $\mathcal{R}$ generates a random $a_i \in \mathbb{Z}_p$, adds the tuple $\langle m_i, a_i \rangle$ to the H_0-list. $\mathcal{R}$ chooses at random $s'_i \in \mathbb{Z}_p$ and returns $(g^{a_i/(x'+s'_i)}, g^{s'_i}, u^{s'_i})$ to $\mathcal{A}$ as the reply for the partial signing query.

It is easy to see that the above Hash queries, partial signing queries and resolution queries are indistinguishably simulated. Finally, $\mathcal{A}$ halts. It either admits failure, in which case so does $\mathcal{R}$, or it returns a full signature σ^* on message m^* without asking the resolution oracle with respect to the message m^* and the challenge signer's public key PK.

By the General Forking Lemma [21] (a standard rewinding technique in random oracle model), with non-negligible probability algorithm $\mathcal{R}$ is able to extracts the value x or x'. The algorithm $\mathcal{R}$ wins if it extracts the value x. Note that the adversary $\mathcal{A}$ can not distinguish between Game 0 and Game 1, because the distributions of the simulation in the two games are the same. Therefore, if $\mathcal{A}$ succeeds with a non-negligible probability ϵ, the algorithm $\mathcal{R}$ wins with a non-negligible probability. □

Theorem 3. *The concrete optimistic fair exchange scheme is secure against the arbitrator in the multi-user setting and chosen-key model if the q-SDH Assumption and q-HSDH assumption hold.*

Proof. Suppose $\mathcal{A}$ makes q partial signing queries. Let m_i be the i-th query submitted by $\mathcal{A}$, $\sigma_P^{(i)} = (\delta_i, v_i, \theta_i)$ be the reply of the partial signing oracle for the i-th query, and s_i be the exponent such that $v_i = g^{s_i}$. Finally the adversary $\mathcal{A}$ outputs a full signature σ^* on message m^* without asking the

partial signing query on message m^*. Let $\sigma_P^* = (\delta^*, v^*, \theta^*) = (\delta^*, g^{s^*}, u^{s^*})$ be the partial signature contained in the full signature σ^*. Below we distinguish two situations.

Type I: There exists $1 \leq i \leq q$ such that $v^* = v_i$, which implies that $s^* = s_i$.

Type II: For all $1 \leq i \leq q$, $v^* \neq v_i$, which implies that $s^* \notin \{s_1, \cdots, s_q\}$.

Type I adversary

Suppose the Type I adversary $\mathcal{A}$ breaks the security against the arbitrator in the multi-user setting and chosen-key model. We show how to construct an algorithm $\mathcal{R}$ to break the q-SDH Assumption. Recall that $\mathcal{R}$ is given $(g, g^x, g^{x^2}, \cdots, g^{x^q})$ as input, its goal is to output $(g^{1/(x+s)}, s)$. $\mathcal{R}$ selects uniformly at random $s_i \in \mathbb{Z}_p$ used for answering the partial signing queries. Let $S = \{s_1, \cdots, s_q\}$ be the set of all s_i, and let $S^i = S \setminus \{s_i\}$. $\mathcal{R}$ also uniformly chooses $i^* \in [q]$. Let $S^* = S \setminus \{s_{i^*}\}$. Define

$$p^*(\eta) = \prod_{t \in S^*}(\eta + t), \quad \text{and} \quad p(\eta) = \prod_{t \in S}(\eta + t).$$

Note that $\deg(p^*) = q - 1$ and $\deg(p) = q$. Define

$$g' = g^{p^*(x)}, h = g^{p(x)}, X = g^x.$$

$\mathcal{R}$ randomly chooses $u \in G$ and sets $\mathsf{PK} := (X, u)$. $\mathcal{R}$ simulates the challenger and answers the queries from $\mathcal{A}$.

H_0 Queries. At any time adversary $\mathcal{A}$ can query the random oracle H_0. Suppose $\mathcal{A}$ makes in total q_0 distinct queries to the random oracle H_0. To response to these queries, $\mathcal{R}$ independently and uniformly chooses $j^* \in [q_0]$ and maintains a list of tuples $\langle m_i', a_i' \rangle$ as explained below. We refer to this list as H_0-list. The list is initially empty. When $\mathcal{A}$ queries the oracle H_0 for the j-th distinct message $m_j' \in \{0,1\}^*$, algorithm $\mathcal{R}$ responds as follows:

1. If the tuple $\langle m_j', a_j' \rangle$ is on the H_0-list, algorithm $\mathcal{R}$ responds with $H(m_j') = h^{a_j'}$ for $j \neq j^*$ and $H(m_{j'}) = g' h^{a_j'}$ for $j = j^*$.
2. Otherwise, if $j \neq j^*$, $\mathcal{R}$ generates a random $a_j' \in \mathbb{Z}_p$, adds the tuple $\langle m_j', a_j' \rangle$ to the H_0-list and responds to $\mathcal{A}$ as $H(m_j) = h^{a_j'}$. If $i = j^*$, $\mathcal{R}$ generates a random $a_{j^*}' \in \mathbb{Z}_p$, adds the tuple $\langle m_{j^*}', a_{j^*}' \rangle$ to the H_0-list and responds to $\mathcal{A}$ as $H(m_{j^*}) = g' h^{a_{j^*}'}$.

H_1 Queries. $\mathcal{A}$ can also query the random oracle H_1. To response to these queries, $\mathcal{R}$ maintains a list of tuples $\langle string^{(i)}, c^{(i)} \rangle$ as explained below. We refer to this list as H_1-list. The list is initially empty. When $\mathcal{A}$ queries the oracle H_1 at a point $string \in \{0,1\}^*$, algorithm $\mathcal{R}$ responds as follows:

1. If the query $string$ already appears on the H-list in some tuple $\langle string, c \rangle$, then algorithm $\mathcal{R}$ responds with $H(string) = c \in \mathbb{Z}_p$.
2. Otherwise, $\mathcal{R}$ generates a random $c \in \mathbb{Z}_p$, adds the tuple $\langle string, c \rangle$ to the H_1-list and responds to $\mathcal{A}$ as $H(string) = c \in \mathbb{Z}_p$.

PSig Queries. Without loss of generality, we assume that for each partial signing query on a message m_i, the adversary has previously asked a H_0 query on m_i. Then for the i-th partial signing query, $\mathcal{R}$ finds the tuple $\langle m'_j, a'_j \rangle$ in the H_0-list such that $m_i = m'_j$. If $m_i = m'_{j^*}$, $\mathcal{R}$ aborts and returns failure. Otherwise $\mathcal{R}$ computes $v_i = g^{s_i}$, $\theta_i = u^{s_i}$ and

$$\delta_i = h^{a'_j/(x+s_i)} = g^{a'_j \prod_{t \in S^i}(x+t)}. \tag{1}$$

Since $g^x, g^{x^2}, \cdots, g^{x^q}$ are known, $\mathcal{R}$ can generate the partial signature $\sigma_P^{(i)} = (\delta_i, v_i, \theta_i)$ without explicitly knowing the secret key x, but instead using the right-hand side of (1) for computing δ_i.

Finally, the adversary $\mathcal{A}$ outputs a full signature σ^* on message m^* without asking the partial signing query on message m^*. From the full signature query, $\mathcal{R}$ can gain a valid partial signature $\sigma_P^* = (\delta^*, v^*, \theta^*) = (\delta^*, g^{s^*}, u^{s^*})$ on message m^*. If $s^* \neq s_{i^*}$ or $m^* \neq m'_{j^*}$, $\mathcal{R}$ returns failure. Otherwise, we show $\mathcal{R}$ can computes a tuple $(g^{1/(x+s^*)}, s^*)$.

Note that

$$\delta^* = (g' h^{a'_{j^*}})^{1/(x+s_{i^*})} = g^{p^*(x)/(x+s_{i^*})} g^{a'_{j^*} p^*(x)}.$$

From δ^* and the knowledge of $g^{a'_{j^*} p^*(x)}$, $\mathcal{R}$ can derive

$$\delta' = (\delta^* / g^{a'_{j^*} p^*(x)}) = g^{p^*(x)/(x+s_{i^*})}.$$

Let $p^*(\eta)/(\eta + s_{i^*}) = p'(\eta) + \gamma_{-1}/(\eta + s_{i^*})$ for some polynomial $p'(\eta)$ of $q-2$ and some $\gamma_{-1} \in \mathbb{Z}_p^*$. Thus $\mathcal{R}$ can further derive

$$\delta'' = (\delta' / g^{p'(x)})^{1/\gamma_{-1}} = (g^{\gamma_{-1}/(x+s_{i^*})})^{1/\gamma_{-1}} = g^{1/(x+s^*)}.$$

Note that i^* and j^* are uniformly chosen and independent of the adversary $\mathcal{A}$'s view, thus the case $s^* = s_{i^*}$ or $m^* = m'_{j^*}$ happens with a non-negligible probability at least $1/qq_0$, which means $\mathcal{R}$ can break the q-SDH assumption.

Type II adversary

Suppose the Type I adversary $\mathcal{A}$ breaks the security against the arbitrator in the multi-user setting and chosen-key model. We show how to construct an algorithm $\mathcal{R}$ to break the q-HSDH Assumption.

Note that algorithm $\mathcal{R}$ is given $\mathbf{Q} =< g, g^x, u, \{(g^{1/(x+s_i)}, g^{s_i}, u^{s_i})\}_{i=1}^{q} >$. Its goal is to output another distinct triple $(g^{1/(x+s)}, g^s, u^s)$. $\mathcal{R}$ forwards $\mathsf{PK} := (g^x, u)$ to $\mathcal{A}$ and simulates the challenger and interacts with adversary $\mathcal{A}$ as follows.

$\mathsf{H_0}$ Queries. At any time adversary $\mathcal{A}$ can query the random oracle H_0. To response to these queries, $\mathcal{R}$ maintains a list of tuples $\langle m_i, a_i \rangle$ as explained below. We refer to this list as H_0-list. The list is initially empty. When $\mathcal{A}$ queries the oracle H_0 on a message $m_i \in \{0,1\}^*$, algorithm $\mathcal{R}$ responds as follows:

1. If the query m_i already appears on the H_0-list in some tuple $\langle m_i, a_i \rangle$, then algorithm $\mathcal{R}$ responds with $H_0(m_i) = g^{a_i} \in \mathbb{Z}_p$.
2. Otherwise, $\mathcal{R}$ generates a random $a_i \in \mathbb{Z}_p$, adds the tuple $\langle m_i, a_i \rangle$ to the H_0-list and responds to $\mathcal{A}$ as $H_0(m_i) = g^{a_i} \in \mathbb{Z}_p$.

$\mathsf{H_1}$ Queries. $\mathcal{A}$ can also query the random oracle H_1. To response to these queries, $\mathcal{R}$ maintains a list of tuples $\langle string^{(i)}, c^{(i)} \rangle$ as explained below. We refer to this list as H_1-list. The list is initially empty. When $\mathcal{A}$ queries the oracle H_1 at a point $string \in \{0,1\}^*$, algorithm $\mathcal{R}$ responds as follows:

1. If the query $string$ already appears on the H-list in some tuple $\langle string, c \rangle$, then algorithm $\mathcal{R}$ responds with $H(string) = c \in \mathbb{Z}_p$.
2. Otherwise, $\mathcal{R}$ generates a random $c \in \mathbb{Z}_p$, adds the tuple $\langle string, c \rangle$ to the H_1-list and responds to $\mathcal{A}$ as $H(string) = c \in \mathbb{Z}_p$.

PSig Queries. Without loss of generality, we assume that for each partial signing query on a message m_i, the adversary has previously asked a H_0 query on m_i. For the i-th partial signing query on message m_i, $\mathcal{R}$ seek out the tuple $\langle m'_j, a'_j \rangle$ in the H_0-list such that $m'_j = m_i$. $\mathcal{R}$ returns $((g^{1/(x+s_i)})^{a'_j}, g^{s_i}, u^{s_i})$ to $\mathcal{A}$ as the reply for the partial signing query.

It is easy to see that the Hash queries and partial signing queries are perfectly simulated. Finally, $\mathcal{A}$ halts. It either admits failure, in which case

so does $\mathcal{R}$, or it returns a full signature σ^* on message m^* without asking the partial signing query on message m^*. From the full signature query, $\mathcal{R}$ can gain a valid partial signature $\sigma_P^* = (\delta^*, v^*, \theta^*) = (\delta^*, g^{s^*}, u^{s^*})$ on message m^*. Due to the randomness of the outputs of H_0 oracle, with overwhelming probability $\mathcal{A}$ have submitted the message m^* to the H_0 oracle. Let $H_0(m^*) = g^a$ where $\langle m^*, a\rangle$ is stored in the H_0-list. Since $s^* \notin \{s_1, \cdots, s_q\}$, $\mathcal{R}$ can simply output $((\delta^*)^{1/a}, v^*, \theta^*)$ and break the q-HSDH assumption.

The theorem follows from the two cases discussed above. The proof is done. □

5.2. An attack in our Enhanced Model

We will show the above concrete construction is insecure in our enhanced model. The reason is that the signing oracle provides valuable information to the adversary. Specifically, we show that if a dishonest arbitrator can have access to the signer's signing oracle, he can gain a partial signature on message $\bar{m}$ from a full signature generated by the signer on message m where $\bar{m}$ is the complement message of m. Since the arbitrator can easily convert a partial signature into a full one, this means the arbitrator is able to generate a full signature on the message $\bar{m}$ without actually asking the corresponding partial one on message $\bar{m}$ from the partial signing oracle.

Recall that a full signature on message m comprises a partial signature on message m, a designated confirmer signature on message $\bar{m}$ with the arbitrator being the designated confirmer, and a zero-knowledge proof of knowledge that can be generated by using either the signer's secret key or the arbitrator's secret key. If the arbitrator observes a full signature generated by the signer on message m, then the arbitrator has a designated confirmer signature on $\bar{m}$. Being the designated confirmer, the arbitrator is able to turn the designated confirmer signature into a conventional signature, which is exactly a partial signature of the optimistic fair exchange scheme on message $\bar{m}$. With the partial signature on message $\bar{m}$ in hand, the arbitrator can further uniformly sample a designated confirmer signature and make a zero-knowledge proof of knowledge by using his own secret key. By doing this, the arbitrator successfully generates a full signature on message $\bar{m}$ without asking the signer to generate the corresponding partial one.

Indeed, the arbitrator can ask the signer to generate a full signature on a message m. Denote the full signature on m is $(\sigma_P, \delta', \gamma', \theta', c_0, t_0, c_1, t_1)$, which is generated by the signer using algorithm Sig. Then the arbitrator

can generater a full signature on message $\bar{m}$ where $\bar{m}$ is the complement message of m as follows.

1. Compute $\sigma_P = (\delta', (\gamma')^{1/y}, \theta')$.
2. Choose at random $s \in \mathbb{Z}_p$, $Z \in G$ and computes $(\delta, \gamma, \theta) := (Z, Y^s, u_i^s)$.
3. Compute $W = e(H_0(m), u_i)/e(\delta, \theta)$.
4. Choose uniformly at random $r, c'_0, t'_0 \in \mathbb{Z}_p$, and computes $c' = H_1(\bar{m}||\mathsf{PK}_i||g^{t'_0}(X_i)^{c'_0}||e(\delta, u_i)^{t'_0}W^{c'_0}||g^r)$, $c'_1 = c' - c'_0 \pmod p$ and $t'_1 = r - c'_1 y \pmod p$.
5. The full signature is set as $\sigma := (\sigma_P, \delta, \gamma, \theta, c'_0, t'_0, c'_1, t'_1)$.

It is straightforward to check that σ is a valid signature on message $\bar{m}$ under the target signer's public key $\mathsf{PK}_i = (X_i, u_i)$.

6. The Insufficiency of Identical Distribution

In this section we show that even the resolved signature generated by the arbitrator has the same distribution with the actual signatures generated by the signer, it still makes a difference whether a malicious arbitrator is allowed to have access to the signing oracle or not. We show this by proposing another concrete counterexample.

The intuition of the concrete optimistic fair exchange scheme is as follows. The partial signature is the BLS signature [22], and the full signature will be the partial signature plus the modified Bender-Katz-Morselli 2-user ring signature between the signer and the arbitrator [23] (which can also be viewed as Waters's signature [24]). Note that in Waters's signature scheme, each user's public key includes g_1, g_2 and a set of Waters hash generators $u', u_1, \cdots, u_n$. In our OFE instantiation, we will let all the users share the same Waters hash generators. Besides, we require that all users use the arbitrator's public key as g_2 so that each user can use a single key to generate both the BLS signature and the modified Bender-Katz-Morselli 2-user ring signature.

Let G be a multiplicative cyclic group of prime order p, g be a generator of G, and $e : G \times G \to G_T$ be a bilinear pairing where G_T is a multiplicative group of order p. Let $H_0 : \{0,1\}^* \to G$ and $H_1 : \{0,1\}^* \to \{0,1\}^n$ be two collision-resistant hash functions. We assume the public parameter (G, p, e, H_1, H_2) is shared by all users. The concrete OFE scheme is as follows.

$\mathsf{Setup}^{\mathsf{TTP}}$: The arbitrator chooses uniformly at random Waters hash generators $u', u_1, \cdots, u_n \leftarrow G$ and exponents $y \leftarrow \mathbb{Z}_p$ and sets $Y = g^y$. APK is set as $(Y, u', u_1, \cdots, u_n)$. ASK is set as y.

$\mathsf{Setup}^{\mathsf{User}}$: User U_i chooses uniformly at random an exponent $x_i \leftarrow \mathbb{Z}_p$ and sets $\mathsf{SK}_i = x_i$ and $\mathsf{PK}_i = g^{x_i}$.

$\mathsf{PSig}(m, \mathsf{SK}_i, \mathsf{APK})$: To partially sign a message, the user U_i computes and sets the partial signature as $\sigma_P = H_0(m)^{x_i}$.

$\mathsf{PVer}(m, \sigma_P, \mathsf{PK}_i, \mathsf{APK})$: Given a partial signature σ_P from user U_i, a verifier verifies whether $e(\sigma_P, g) = e(H_0(m), \mathsf{PK}_i)$ holds. If so, it returns $\top$; otherwise it returns $\perp$.

$\mathsf{Sig}(m, \mathsf{SK}_i, \mathsf{APK})$: To sign a message, the user U_i computes $\sigma_P \leftarrow \mathsf{PSig}(m, \mathsf{SK}_i, \mathsf{APK})$ and $(m_1, \cdots, m_n) \leftarrow H_1(m||\mathsf{PK}_i)$. It then chooses $r \leftarrow \mathbb{Z}_p$ and computes

$$S_1 = Y^{x_i} \cdot (u' \prod_{j=1}^{n} u_j^{m_j})^r, \quad \text{and} \quad S_2 = g^r.$$

The full signature is set as $\sigma = (\sigma_P, S_1, S_2)$.

$\mathsf{Ver}(m, \sigma_P, \mathsf{PK}_i, APK)$: Given a full signature σ from user U_i, a verifier verifies whether $\mathsf{PVer}(m, \sigma_P, \mathsf{PK}_i, \mathsf{APK}) = \top$ and whether

$$e(Y, \mathsf{PK}_i) = e(S_1, g) \cdot e(S_2^{-1}, u' \prod_{j=1}^{n} u_j^{m_j}).$$

If both equations hold, it returns $\top$; otherwise, it returns $\perp$.

$\mathsf{Res}(m, \sigma_P, \mathsf{ASK}, \mathsf{PK}_i)$: It first verifies whether σ_P is a valid partial signature by running $\mathsf{PVer}(m, \sigma_P, \mathsf{PK}_i, \mathsf{APK})$. If σ_P is invalid, it returns $\perp$. Otherwise, it computes $(m_1, \cdots, m_n) \leftarrow H_1(m||\mathsf{PK}_i)$, chooses $r \leftarrow \mathbb{Z}_p$, computes

$$S_1 = \mathsf{PK}_i^y \cdot (u' \prod_{j=1}^{n} u_j^{m_j})^r, \quad \text{and} \quad S_2 = g^r,$$

and returns $\sigma = (\sigma_P, S_1, S_2)$.

It is not hard to verify that in the above construction, any (partial) signature created by Sig (PSig) will be valid under Ver (PVer), and that any signature created by the arbitrator using Res based on a partial signature

generated by PSig will be valid under Ver. Thus the correctness property of the above construction holds.

It is easy to see the actual signatures generated by the signer and the resolved signatures generated by the arbitrator have an identical distribution, thus the resolution ambiguity also holds.

6.1. Security Analysis

We show the specific construction is secure in the multi-user setting and chosen-key model reviewed in Section 3.1. Intuitively, the security against signers holds unconditionally as the arbitrator is always able to convert a signer's partial signature into a full one. The security against verifiers holds due to unforgeability of the BLS signature and the modified Bender-Katz-Morselli 2-user ring signature. The security against the arbitrator holds due to the unforgeability of the BLS signature.

Theorem 4. *The concrete optimistic fair exchange scheme above is unconditionally secure against signers in the multi-user setting and chosen-key model.*

Proof. Obviously, for any message m and any valid signature σ_P on m under the verification key PK_i, the arbitrator can always convert it into a full signature by using its own secret key ASK. Therefore, the security against signers always hold. □

Theorem 5. *The concrete optimistic fair exchange scheme above is secure against verifiers in the multi-user setting and chosen-key model if CDH assumption holds in G.*

Proof. Suppose an adversary $\mathcal{A}$ breaks the security against verifiers with non-negligible probability. We show how to construct an algorithm $\mathcal{R}$ that breaks the CDH assumption in G.

Note that algorithm $\mathcal{R}$ is given an CDH instance $(G, p, g, A = g^a, B = g^b)$. Its goal is to output g^{ab}. Algorithm $\mathcal{R}$ simulates the challenger and interacts with adversary $\mathcal{A}$. Let q be the number of different messages contained in the queries $\mathcal{A}$ has made to the resolution oracle.

At the start, the algorithm $\mathcal{R}$ sets an integer, $l = 4q$, and chooses uniformly at random an integer, k^* between 0 and n. It then chooses an n-length vector, $\vec{x} = (x_i)$, where the elements of $\vec{x}$ are chosen uniformly at random between 0 and $l-1$ and a value, x', chosen uniformly at random between 0 and

$l-1$. Besides, algorithm $\mathcal{R}$ chooses a random $y' \in \mathbb{Z}_p$ and an n-length vector, $\vec{y} = (y_i)$, where the elements of $\vec{y}$ are chosen at random in $\mathbb{Z}_p$. Algorithm $\mathcal{R}$ keeps these values private. Let $e : G \times G \to G_T$ be a bilinear pairing where G_T is a multiplicative group of order p and $H_1 : \{0,1\}^* \to \{0,1\}^n$ be a collision-resistant hash function. Algorithm $\mathcal{R}$ sets $\mathsf{PK} := A$ and $\mathsf{APK} = Y := B$ and assigns the Waters hash generators parameters $u' = B^{p-k^*m+x'}g^{y'}$ and $u_i = B^{x_i}g^{y_i}$. Algorithm $\mathcal{R}$ forwards PK, APK and the public parameters $(G, p, e, H_0, H_1, u', u_1, \cdots, u_n)$ to the adversary. From the view of the adversary, the distribution of the simulated public parameters is identical to the real construction. We view the hash function H_0 as a random oracle.

For ease of analysis we define the following functions where $\tilde{m}$ is the set of indices i such that $m_i = 1$ while $H_1(m) = (m_1, \cdots, m_n)$:

$$F(m) = (p - mk^*) + x' + \sum_{i \in \tilde{m}} x_i, \quad \text{and} \quad J(m) = y' + \sum_{i \in \tilde{m}} y_i.$$

We also define a binary function $K(m)$ as

$$K(m) = \begin{cases} 0, & \text{if } \; x' + \sum_{i \in \tilde{m}} x_i \equiv 0 \pmod{l} \\ 1, & \text{otherwise} \end{cases}.$$

$\mathsf{H_0}$ Queries. At any time adversary $\mathcal{A}$ can query the random oracle H_0. To response to these queries, $\mathcal{R}$ maintains a list of tuples $\langle M_i, a_i \rangle$ as explained below. We refer to this list as H_0-list. The list is initially empty. When $\mathcal{A}$ queries the oracle H_0 on a message $M_i \in \{0,1\}^*$, algorithm $\mathcal{R}$ responds as follows:

1. If the query M_i already appears on the H_0-list in some tuple $\langle M_i, a_i \rangle$, then algorithm $\mathcal{R}$ responds with $H(M_i) = g^{a_i}$.
2. Otherwise, $\mathcal{R}$ generates a random $a_i \in \mathbb{Z}_p$, adds the tuple $\langle M_i, a_i \rangle$ to the H_0-list and responds to $\mathcal{A}$ as $H(M_i) = g^{a_i}$.

PSig Queries. For the i-th partial signing query on message M_i, $\mathcal{R}$ checks whether there is a tuple $\langle M_i, a_i \rangle$ in the H_0-list. If not, $\mathcal{R}$ generates a random $a_i \in \mathbb{Z}_p$, adds the tuple $\langle M_i, a_i \rangle$ to the H_0-list. $\mathcal{R}$ returns A^{a_i} to $\mathcal{A}$ as the reply for the partial signing query.

Res Queries. Given a resolution query $(m, \sigma_P, \mathsf{PK}_i)$ where PK_i could be PK or could be adversarially-generated by the adversary, algorithm $\mathcal{R}$ responds to this query as follows:

1. checks whether $\mathsf{PVer}(\sigma_P, m, \mathsf{PK}_i) = \top$. If so, continues; otherwise, $\mathcal{R}$ responds to $\mathcal{A}$ with a special symbol $\perp$.
2. chooses a random $r \in \mathbb{Z}_p$, and computes the signature as

$$S_1 = g_1^{\frac{-J(m||\mathsf{PK}_i)}{F(m||\mathsf{PK}_i)}}(u' \prod_{i \in m||\tilde{\mathsf{PK}}_i} u_i)^r, \quad S_2 = g_1^{\frac{-1}{F(m||\mathsf{PK}_i)}} g^r.$$

3. The full signature is set as $\sigma := (\sigma_P, S_1, S_2)$ and returned to $\mathcal{A}$ as the reply of the resolution query.

Let $\tilde{r} = r - \frac{\alpha}{F(m||\mathsf{PK}_i)}$ and $\mathsf{PK}_i = g^\alpha$. Then we have

$$\begin{aligned}
S_1 &= \mathsf{PK}_i^{\frac{-J(m||\mathsf{PK}_i)}{F(m||\mathsf{PK}_i)}}(u' \textstyle\prod_{i \in m||\tilde{\mathsf{PK}}_i} u_i)^r \\
&= \mathsf{PK}_i^{\frac{-J(m||\mathsf{PK}_i)}{F(m||\mathsf{PK}_i)}}(B^{F(m||\mathsf{PK}_i)} g^{J(m||\mathsf{PK}_i)})^r \\
&= B^\alpha (B^{F(m||\mathsf{PK}_i)} g^{J(m||\mathsf{PK}_i)})^{-\frac{\alpha}{F(m||\mathsf{PK}_i)}} (B^{F(m||\mathsf{PK}_i)} g^{J(m||\mathsf{PK}_i)})^r \\
&= B^\alpha (u' \textstyle\prod_{i \in m||\tilde{\mathsf{PK}}_i} u_i)^{r - \frac{\alpha}{F(m||\mathsf{PK}_i)}} \\
&= B^\alpha (u' \textstyle\prod_{i \in m||\tilde{\mathsf{PK}}_i} u_i)^{\tilde{r}}
\end{aligned}$$

Additionally, we have

$$S_2 = g_1^{\frac{-1}{F(m||\mathsf{PK}_i)}} g^r = g^{r - \frac{\alpha}{F(m||\mathsf{PK}_i)}} = g^{\tilde{r}}.$$

Algorithm $\mathcal{R}$ will be able to perform this computation if and only if $F(m||\mathsf{PK}_i) \neq 0 \mod p$.

Finally the adversary $\mathcal{A}$ halts. It either admits failure, in which case so does $\mathcal{R}$, or it returns a full signature $\sigma^* = (\sigma_P^*, S_1^*, S_2^*)$ on message m^* without asking the resolution oracle with respect to the message m^* and the challenge signer's public key PK. Due to the collision-resistant property of the hash function, $H_1(m^*||\mathsf{PK}) \neq H(m||\mathsf{PK}_i)$ for any message m and PK_i that had been submitted to the resolution query before. Let $m^*\tilde{||}\mathsf{PK}$ be the set of indices i such that $m_i^* = 1$ where $H(m^*||\mathsf{PK}) = (m_1^*, \cdots, m_n^*)$. If $x' + \sum_{i \in m^*\tilde{||}\mathsf{PK}} x_i = k^* m$, then we have

$$e(\frac{S_1^*}{(S_2^*)^{y + \sum_{i \in m^*\tilde{||}\mathsf{PK}} y_i}}, g) = \frac{e(A, B) e(S_2^*, u' \prod_{j=1}^n u_j^{m_j^*})}{e((S_2^*)^{y + \sum_{i \in m^*\tilde{||}\mathsf{PK}} y_i}, g)} = e(A, B).$$

Therefore algorithm $\mathcal{R}$ can solve the CDH problem by computing g^{ab} as

$$g^{ab} = \frac{S_1^*}{(S_2^*)^{y+\sum_{i\in m^*\tilde{\|}\mathsf{PK}} y_i}}.$$

Similar to [24], the probability that the simulator does not abort during simulating the signing oracle and the equation $x' + \sum_{i\in m^*\tilde{\|}\mathsf{PK}} x_i = k^* m$ holds is at least $\lambda = \frac{1}{8(n+1)q}$, which is non-negligible. This completes the proof. □

Theorem 6. *The concrete optimistic fair exchange scheme above is secure against the arbitrator in the multi-user setting and chosen-key model if the CDH assumption holds in G.*

Proof. Suppose an adversary breaks the security against verifiers with non-negligible probability. We show how to construct an algorithm $\mathcal{R}$ that breaks the CDH assumption in G.

Note that algorithm $\mathcal{R}$ is given an CDH instance $(G, p, g, A = g^a, B = g^b)$. Its goal is to output g^{ab}. The algorithm $\mathcal{R}$ chooses uniformly at random $u', u_1, \cdots, u_n \in G$. Let $e : G \times G \to G_T$ be a bilinear pairing where G_T is a multiplicative group of order p and $H_1 : \{0,1\}^* \to \{0,1\}^n$ be a collision-resistant hash function. Algorithm $\mathcal{R}$ sets $\mathsf{PK} := A$ and forwards PK and the public parameters $(G, p, e, H_0, H_1, u', u_1, \cdots, u_n)$ to the adversary $\mathcal{A}$. From the view of the adversary, the distribution of the simulated public parameters is identical to the real construction. We view the hash function H_0 as a random oracle. Suppose the adversary $\mathcal{A}$ makes in total q different queries to the random oracle. At the start, the algorithm $\mathcal{R}$ chooses uniformly at random an integer k^* such that $1 \leq k^* \leq q$.

$\mathsf{H_0}$ Queries. At any time adversary $\mathcal{A}$ can query the random oracle H_0. To response to these queries, $\mathcal{R}$ maintains a list of tuples $\langle M_i, a_i \rangle$ as explained below. We refer to this list as H_0-list. The list is initially empty. When $\mathcal{A}$ makes the i different queries the oracle H_0 on a message $M_i \in \{0,1\}^*$, algorithm $\mathcal{R}$ responds as follows:

1. $\mathcal{R}$ checks whether $i = k^*$. If so, $\mathcal{R}$ returns B to $\mathcal{A}$ as the reply and adds the tuple $\langle M_i, B \rangle$ to the H_0-list.
2. Otherwise, $\mathcal{R}$ generates a random $a_i \in \mathbb{Z}_p$, adds the tuple $\langle M_i, a_i \rangle$ to the H_0-list and responds to $\mathcal{A}$ as $H(M_i) = g^{a_i}$.

PSig Queries. Without loss of generality, we assume that for each partial signing query, the adversary $\mathcal{A}$ has previously submitted the corresponding message to the H_0 query. For a partial signing query on message M, $\mathcal{R}$ checks the H_0-list. Suppose $M = M_i$ where M_i is the i-th different query $\mathcal{A}$ has made to the H_0 oracle. If $i = k^*$, $\mathcal{R}$ aborts and returns failure. Otherwise, $\mathcal{R}$ returns A^{a_i} to $\mathcal{A}$ as the reply for the partial signing query.

Finally $\mathcal{A}$ outputs (m^*, σ^*) such that m^* is not queried to the partial signing oracle. Since k^* is chosen uniformly at random and independent of the adversary $\mathcal{A}$'s behavior, thus with non-negligible probability at least $1/q$ we have $m^* = M_{k^*}$, in which case $\sigma^* = g^{ab}$. Thus algorithm $\mathcal{R}$ can outputs σ^* and breaks the CDH assumption with non-negligible probability. □

6.2. An attack in our Enhanced Model

The malicious arbitrator can always sign on behalf the signer as follows when it is provided the signing oracle. The reason that an adversary can succeed in our enhanced model is that the signing oracle can provide useful information to the adversary. Specifically, the adversary can forge a new signature as follows.

1. The arbitrator computes $Y = H_0(m^*)$ where m^* is the message the arbitrator tries to forge a signature on. Besides, the arbitrator chooses uniformly at random $x', x_1, \cdots, x_n \in \mathbb{Z}_p$ and sets $u' = g^{x'}, u_1 = g^{x_1}, \cdots, u_n = g^{x_n}$. The public key APK is set as $(Y, u', u_1, \cdots, u_n)$. Note that APK is malicious generated in the sense that the arbitrator does not know the corresponding secret key of Y.
2. Let the challenger's public key is $X = g^x$. The arbitrator asks a signing query on a message $m \neq m^*$, and gains a ring signature

$$S_1 = Y^x \cdot (u' \prod_{j=1}^{n} u_j^{m_j})^r, \quad \text{and} \quad S_2 = g^r,$$

   where $(m_1, \cdots, m_n) \leftarrow H_1(m||X)$.
3. With the knowledge of $x', x_1, \cdots, x_n$, the arbitrator computes

$$A = (S_2)^{x'} \prod_{j=1}^{n} (S_2)^{x_j m'_j}$$

   and therefore $Y^x = S_1/A$.

4. The arbitrator sets $\sigma_P := Y^x$, chooses at random $r' \in \mathbb{Z}_p$, and computes
$$S_1' = Y^x \cdot (u' \prod_{j=1}^{n} u_j^{m_j'})^{r'}, \quad \text{and} \quad S_2' = g^{r'},$$
where $(m_1', \cdots, m_n') \leftarrow H_1(m^*||X)$.
5. The full signature is set as $\sigma := (\sigma_P, S_1', S_2')$.

It is not hard to check that σ is a valid signature on message m^* under the target signer's public key X and the arbitrator has not makes a partial signing query on the message m^*.

7. Previous Paradigms Revisited

In this section, we address the basic theoretical question, namely whether or not a scheme satisfying the security notions exists. We revisit two previous known paradigms for constructing optimistic fair exchange schemes and evaluate their security in our enhanced model.

7.1. Verifiably Encrypted Signature Paradigm

Dodis, Lee and Yum [10] showed optimistic fair exchange schemes secure in the multi-user setting can be constructed from verifiably encrypted signatures. Subsequently, Huang et al. [11] showed this generic construction also works in the chosen-key model. The generic construction of optimistic fair exchange schemes based on verifiably encrypted signatures [10] is as follows.

Let $\mathcal{E}$ = (KGen, Enc, Dec) be an encryption scheme, and $\mathcal{S}$ = (KGen, Sig, Ver) be a signature scheme. Given an encryption key pair (ek, dk) and a signature key pair (sk, vk), we let Π be a simulation-sound [25] non-interactive zero-knowledge (NIZK) proof system for the NP-language $L = \{(c, m, ek, vk) | \exists \sigma \ [c = \mathcal{E}.\mathsf{Enc}(\sigma, ek) \wedge \mathcal{S}.\mathsf{Ver}(m, \sigma, vk) = 1]\}$. Intuitively, the partial signature of the generic optimistic fair exchange scheme is the encryption of the signer's signature plus a non-interactive zero-knowledge proof showing that the ciphertext is correctly generated. The full signature is the signer's signature. Below is the generic construction of optimistic fair exchange schemes from verifiably encrypted signatures.

- $\mathsf{Setup}^{\mathsf{TTP}}$: The arbitrator runs $(ek, dk) \leftarrow \mathcal{S}.\mathsf{KGen}(1^k)$ and sets $\mathsf{ASK} = dk$ and $\mathsf{APK} = ek$.

- $\mathsf{Setup}^{\mathsf{User}}$: Each user U_i runs $(sk_i,\ vk_i) \leftarrow \mathcal{S}.\mathsf{KGen}(1^k)$ and sets $(\mathsf{SK}_i,\ \mathsf{PK}_i) = (sk_i,\ vk_i)$.
- Sig: To sign a message m, the user U_i runs $s \leftarrow \mathcal{S}.\mathsf{Sig}(sk_i, m)$. The full signature is set as $\sigma = s$.
- Ver: To verify whether a full signature σ is generated by user U_i, a verifier checks whether the output of $\mathcal{S}.\mathsf{Ver}(m, \sigma, vk_i)$ is valid or not. If valid, return $\top$. Otherwise, return $\bot$.
- PSig: To partially sign a message m, the user U_i runs $s \leftarrow \mathcal{S}.\mathsf{Sig}(sk_i, m)$ and $c \leftarrow \mathcal{E}.\mathsf{Enc}(s, ek)$ and generates a proof π showing $(c, m, ek, vk_i) \in L$. The partial signature is set as $\sigma_P = (c, \pi)$.
- PVer: To verify whether a partial signature $\sigma_P = (c, \pi)$ is generated by user U_i on message m, the verifier checks whether the proof π for the statement $(c, m, ek, vk_i) \in L$ is valid or not. If valid, return $\top$. Otherwise, return $\bot$.
- Res: For a partial signature $\sigma_P = (c, \pi)$ generated by the user U_i on message m, the arbitrator checks whether $\mathsf{PVer}(m, \sigma_P, \mathsf{ASK}, \mathsf{PK}_i) = \top$. If so, the arbitrator runs $s \leftarrow \mathcal{E}.\mathsf{Dec}(dk, c)$ and returns $\sigma = s$. Otherwise, the arbitrator returns $\bot$.

The correctness property of this construction follows from the correctness of the signature scheme $\mathcal{S}$, the correctness of the encryption scheme $\mathcal{E}$, and the completeness of the NIZK proof system Π, which states that if the statement is true, the honest verifier will be convinced of this fact by an honest prover.

The resolution ambiguity property holds due to the fact that a partial signature is a verifiably encryption of a full signature and the resolution process is just a decryption of the partial signature.

For the security analysis, the simulation-sound property [25] is required. Intuitively, the simulation-sound property states that a probabilistic polynomial time adversary is incapable of convincing others of a false statement even after seeing a simulated proof of its choice. We have the following theorem.

Theorem 7. *If $\mathcal{S}$ is* UF-CMA*-secure signature scheme [26], $\mathcal{E}$ is a* CCA2*-secure encryption scheme [27], and Π is a simulation-sound NIZK proof system, then the above optimistic fair exchange scheme constructed from verifiably encrypted signatures is secure in our enhanced model.*

Proof. Security against Signers. The proof about security against signers is the same as that in [10], and therefore we omit it.

Security against verifiers. Suppose an adversary $\mathcal{A}$ breaks the security against verifiers by outputting (m^*, σ^*). If $\mathcal{A}$ has not made a partial signing query on message m^*, the analysis of this type of attack is covered in the security against the arbitrator to be discussed later. Thus without loss of generality, we assume that $\mathcal{A}$ has made the query m^* to the partial signing oracle O_{PSig}.

In this setting, we show how to construct an algorithm $\mathcal{B}$ that breaks the $\mathsf{CCA2}$-security of the encryption scheme $\mathcal{E}$. Recall that $\mathcal{B}$ gets ek as input and has access to the decryption oracle O_{Dec}. $\mathcal{B}$ runs $(sk, vk) \leftarrow \mathcal{S}.\mathsf{KGen}(1^k)$ and sets $\mathsf{PK} = vk$, $\mathsf{APK} = ek$. $\mathcal{B}$ forwards PK and APK to $\mathcal{A}$ and simulates the oracles for $\mathcal{A}$.

Let q be the total number of O_{PSig} queries made by $\mathcal{A}$. To reply to the i-th query m_i to the partial signing oracle O_{PSig}, $\mathcal{B}$ uniformly chooses j from $\{1, 2, \cdots, q\}$ and does as follows.

- If $i = j$, $\mathcal{B}$ randomly chooses a message $\bar{m}_0$, and computes $\bar{s}_0 = \mathcal{S}.\mathsf{Sig}(sk, \bar{m}_0)$ and $\bar{s}_1 = \mathcal{S}.\mathsf{Sig}(sk, m_j)$. $\mathcal{B}$ submits $(\bar{s}_0, \bar{s}_1)$ to its own challenger. Let $\bar{c}_b$ be the challenge ciphertext returned by $\mathcal{B}$'s own challenger, which is either $\mathcal{E}.\mathsf{Enc}(\bar{s}_0, ek)$ or $\mathcal{E}.\mathsf{Enc}(\bar{s}_1, ek)$. $\mathcal{B}$ further simulates a proof π_j showing $(\bar{c}_b, m_j, ek, vk) \in L$. $\mathcal{B}$ forwards $(\bar{c}_b, \pi_j)$ to $\mathcal{A}$ the reply of this query.

- If $i \neq j$, $\mathcal{B}$ computes $s_i = \mathcal{S}.\mathsf{Sig}(sk, m_i)$, $c_i = \mathcal{E}.\mathsf{Enc}(s_i, ek)$ and generates a proof showing that $(c_i, m_i, ek, vk) \in L$. The reply of this query is (c_i, π_i).

To simulate a query m to the signing oracle O_{Sig}, $\mathcal{B}$ computes $s = \mathcal{S}.\mathsf{Sig}(sk, m)$ and returns s as the reply.

To simulate a query $(m, (c, \pi), \mathsf{PK}_i)$ to the resolution oracle O_{Res}, $\mathcal{B}$ does as follows.

- If π is valid and $c \neq \bar{c}_b$, $\mathcal{B}$ submits c to its own decryption oracle O_{Dec}. Let the reply be s. $\mathcal{B}$ forwards s to $\mathcal{A}$ as the reply.

- If π is valid and $c = \bar{c}_b$. $\mathcal{B}$ further checks whether $m = m_j$, $\mathsf{PK}_i = \mathsf{PK}$ and $\pi = \pi_j$. If so, $\mathcal{B}$ aborts and outputs a random bit to its CCA challenger. Otherwise, by the simulation-sound property of Π, the decryption of c must be a valid signature on message m under the

public key PK_i. Suppose $\mathcal{S}.\mathsf{Ver}(m, \bar{s}_b, \mathsf{PK}_i) = \top$ where $b \in \{0, 1\}$. $\mathcal{B}$ returns $\bar{s}_b$ to $\mathcal{A}$ and at the same time outputs the bit b to its own CCA challenger.

- If π is invalid, $\mathcal{B}$ returns $\perp$.

Note that if the challenge ciphertext $\bar{c}_b$ is the encryption of $\bar{s}_0$ (i.e., b=0), c_j has no information on the signature of m_j and $\mathcal{A}$'s chance of outputting a signature on message m_j under PK is negligible. If the challenge ciphertext is the encryption of $\bar{s}_1$ (i.e., b=1), then the simulated environment is indistinguishable with that in the real attack environment and with non-negligible probability $\mathcal{A}$ outputs a signature on message m_j. Thus if $\mathcal{A}$'s final output (m^*, σ^*) satisfy $m^* = m_j$, $\mathcal{B}$ outputs 1 and otherwise $\mathcal{B}$ outputs a random bit to its CCA challenger. It is easy to see that if $\mathcal{A}$ can forge a full signature with non-negligible probability, then $\mathcal{B}$ can break the $\mathsf{CCA2}$-security of the encryption scheme $\mathcal{E}$ with at least non-negligible probability.

Security against the arbitrator. To show security against the arbitrator, we convert any arbitrator $\mathcal{A}$ that breaks the security against the arbitrator into an adversary $\mathcal{B}$ that forges a new signature for the underlying signature scheme $\mathcal{S}$. The adversary $\mathcal{B}$, on input vk, sets $\mathsf{PK} = vk$, receives APK from $\mathcal{A}$ and simulates the environments for $\mathcal{A}$. To respond to a partial signing query on message m, $\mathcal{B}$ asks a signature s from its own signing oracle on message m, computes $c = \mathcal{E}.\mathsf{Enc}(s, \mathsf{APK})$ and generates a proof showing $(c, m, \mathsf{APK}, vk) \in L$. To respond to a signing query on message m, $\mathcal{B}$ asks a signature s from its own signing oracle on message m and returns s to $\mathcal{A}$ as the reply. The simulation is perfect and finally $\mathcal{A}$ outputs (m^*, σ^*) such that m^* is not an input to the partial signing oracle or the signing oracle. Thus $\mathcal{B}$ can simply win by outputting (m^*, σ^*). □

7.2. Conventional Signature and Ring signature Paradigm

Since the optimistic fair exchange schemes resulted from verifiably encrypted signatures may not be efficient in the standard model due to the involvement of a simulation-sound proof, Huang et al. [11] suggested another generic methodology for constructing optimistic fair exchange schemes secure in the multi-user setting and chosen-key model. The generic construction is built on conventional signatures and ring signature.

Let $\mathcal{S} = (\mathsf{KGen}, \mathsf{Sig}, \mathsf{Ver})$ be a conventional signature scheme, and $\mathsf{RS} = (\mathsf{KGen}, \mathsf{Sig}, \mathsf{Ver})$ be a ring signature scheme. In Huang et al.'s generic construction, each signer has two key pairs, one for the conventional signature

scheme $\mathcal{S}$ and the other for the ring signature scheme RS. The arbitrator has only a key pair for the ring signature scheme RS. The partial signature is a conventional signature σ' generated by the signature scheme $\mathcal{S}$. The full signature σ is the partial signature σ' plus a 2-user ring signature σ_r generated by the ring signature scheme RS with the ring members consist of the signer and the arbitrator. To convert a partial signature into a full one, the arbitrator simply produces a ring signature σ_r using its secret key with the ring numbers being the signer and the arbitrator. We review the construction as follows.

- $\mathsf{Setup}^{\mathsf{TTP}}$: The arbitrator runs $(ask, apk) \leftarrow \mathsf{RS.KGen}(1^k)$ and sets $(\mathsf{ASK}, \mathsf{APK}) := (ask, apk)$.
- $\mathsf{Setup}^{\mathsf{User}}$: Each user U_i runs $(sk_i, pk_i) \leftarrow \mathcal{S}.\mathsf{KGen}(1^k)$, $(rsk_i, rpk_i) \leftarrow \mathsf{RS.KGen}(1^k)$, and sets $(\mathsf{SK}_i, \mathsf{PK}_i) := ((sk_i, rsk_i), (pk_i, rpk_i))$.
- Sig: To sign a full signature on message m, user U_i runs $\sigma' \leftarrow \mathcal{S}.\mathsf{Sig}(sk_i, m)$ and $\sigma_r \leftarrow \mathsf{RS.Sig}(rsk_i, m||\mathsf{PK}_i, R)$[2] where $R := \{rpk_i, apk\}$. The full signature is then set as $\sigma := (\sigma', \sigma_r)$.
- Ver: To verify whether a full signature $\sigma := (\sigma', \sigma_r)$ is generated by user U_i on message m, the verifier checks whether both $\mathcal{S}.\mathsf{Ver}(m, \sigma', pk_i) = \top$ and $\mathsf{RS.Ver}(m||\mathsf{PK}_i, \sigma_r, R) = \top$ where $R := \{rpk_i, apk\}$. If so, it returns $\top$; otherwise, it returns $\bot$.
- PSig: To partially sign a full signature on message m, user U_i runs $\sigma' \leftarrow \mathcal{S}.\mathsf{Sig}(sk_i, m)$, and returns σ' as the partial signature.
- PVer: To verify whether a partial signature $\sigma_P := \sigma'$ is generated by user U_i on message m, a verifier returns $\mathcal{S}.\mathsf{Ver}(m, \sigma', pk_i)$.
- Res: For a partial signature $\sigma_P := \sigma'$ generated by the user U_i on message m, the arbitrator first checks the validity of σ' by running $\mathcal{S}.\mathsf{Ver}(m, \sigma', pk_i)$. If σ' is invalid, it returns $\bot$; otherwise, it computes $\sigma_r \leftarrow \mathsf{RS.Sig}(ask, m||\mathsf{PK}_i, R)$, where $R := \{rpk_i, apk\}$. The arbitrator returns $\sigma := (\sigma', \sigma_r)$.

[2]In Huang et al.'s construction, the ring signature is computed as $\sigma_r \leftarrow \mathsf{RS.Sig}(rsk_i, m||\sigma'||\mathsf{PK}_i, R)$. We will later show it would suffice even if we sign on a shorter message string $m||\mathsf{PK}_i$ rather than $m||\sigma'||\mathsf{PK}_i$.

The correctness property of the construction from this paradigm holds due to correctness of the conventional signature scheme $\mathcal{S}$ and the ring signature scheme RS. The resolution ambiguity property follows the anonymity requirement of the ring signature scheme RS. For the security analysis, we require the ring signature to be unforgeable under an adaptive attack against a static adversary, a ring signature model which was proposed in [28].

Theorem 8. *If $\mathcal{S}$ is a conventional signature scheme that is* $\mathsf{UF\text{-}CMA}$*-secure, and* RS *is a secure ring signature scheme with basic anonymity and existential unforgeability under an adaptive attack against a static adversary, then the above optimistic fair exchange scheme constructed from conventional signatures and ring signatures is secure in our enhanced model.*

Proof. Security against signers. The proof about security against signers is the same as that in [10], and therefore we omit it.

Security against verifiers. Suppose that an adversary $\mathcal{A}$ breaks the security against verifiers. We show how to construct an algorithm $\mathcal{B}$ that breaks the unforgeability of RS under an adaptive attack against a static adversary.

Given two public keys pk_0 and pk_1, which are challenge public keys, $\mathcal{B}$ runs $(sk, pk) \leftarrow \mathcal{S}.\mathsf{KGen}(1^k)$, and sets $\mathsf{APK} := pk_1$ and $\mathsf{PK} := (pk, pk_0)$. $\mathcal{B}$ forwards to $\mathcal{A}$ $(\mathsf{PK}, \mathsf{APK})$ and simulates the oracles for $\mathcal{A}$.

When $\mathcal{A}$ makes a partial signing query m to oracle O_{PSig}, $\mathcal{B}$ computes and returns $\mathcal{S}.\mathsf{Sig}(sk, m)$ to $\mathcal{A}$.

When $\mathcal{A}$ makes a signing query m to oracle O_{PSig}, $\mathcal{B}$ firstly runs $\sigma' \leftarrow \mathcal{S}.\mathsf{Sig}(sk, m)$ and gains a ring signature σ_r on message $m||\mathsf{PK}$ under the ring $\{pk_0, pk_1\}$ generated using the secret key corresponding to pk_0 from its own ring signing oracle. $\mathcal{B}$ forwards (σ', σ_r) to $\mathcal{A}$ as the reply.

When $\mathcal{A}$ makes a resolution query $(m, \sigma', \mathsf{PK}'_i)$ where $\mathsf{PK}'_i := (pk'_i, rpk'_i)$ to oracle O_{Res}, $\mathcal{B}$ checks whether $\mathsf{PVer}(m, \sigma', \mathsf{PK}'_i, \mathsf{APK}) = \top$. If not, $\mathcal{B}$ returns $\perp$ to $\mathcal{A}$. Otherwise, $\mathcal{B}$ gains a ring signature σ_r on message $m||\mathsf{PK}'_i$ under the ring $\{rpk'_i, pk_1\}$ generated using the secret key corresponding to pk_1 from its own ring signing oracle. $\mathcal{B}$ forwards (σ', σ_r) to $\mathcal{A}$.

The simulation is perfect. Finally, $\mathcal{A}$ outputs its forgery $(\tilde{m}, \tilde{\sigma})$, where $\tilde{\sigma} = (\tilde{\sigma}', \tilde{\sigma}_r)$. Thus we have $\mathsf{RS.Ver}(\tilde{m}||\mathsf{PK}, \tilde{\sigma}_r, R) = \top$ where $R := \{pk_0, pk_1\}$. Since $(\tilde{m}, \cdot, \mathsf{PK}) \notin Query(\mathcal{A}, O_{\mathsf{Sig}})$ and $(\tilde{m}, \cdot, \mathsf{PK}) \notin Query(\mathcal{A}, O_{\mathsf{Res}})$, $\mathcal{B}$ has never issued the query on message $\tilde{m}||\mathsf{PK}$ under the ring $\{pk_0, pk_1\}$ to its own ring signing oracle. Therefore $\tilde{\sigma}_r$ is a valid ring signature on a new message $\tilde{m}||\mathsf{PK}$ under the ring $\{pk_0, pk_1\}$. $\mathcal{B}$ can simply output $(\tilde{m}||\mathsf{PK}, \tilde{\sigma}_r)$

and break the existential unforgeability under an adaptive attack against a static adversary.

Security against the arbitrator. Suppose that an adversary $\mathcal{A}$ breaks the security against the arbitrator. We show how to construct an algorithm $\mathcal{B}$ that breaks the unforgeability of $\mathcal{S}$. Recall that $\mathcal{B}$ gets vk as input and has access to its own signing oracle O_{sk}. $\mathcal{B}$ runs $(rsk, rpk) \leftarrow \mathsf{RS.KGen}(1^k)$ and forwards $\mathsf{PK} := (vk, rpk)$ to $\mathcal{A}$, who returns to $\mathcal{B}$ the public arbitration key APK. To respond to a partial signing query on message m, $\mathcal{B}$ asks a signature s from its own signing oracle O_{sk} on message m, and returns s to $\mathcal{A}$ as the reply. To respond to a signing query on message m, $\mathcal{B}$ asks a signature s from its own signing oracle on message m, runs $\sigma_r \leftarrow \mathsf{RS.Sig}(rsk, m||\mathsf{PK}, R)$ where $R = \{rpk, \mathsf{APK}\}$ and returns (s, σ_r) to $\mathcal{A}$ as the reply. The simulation is perfect and finally $\mathcal{A}$ outputs $(\tilde{m}, \tilde{\sigma})$, where $\tilde{\sigma} = (\tilde{\sigma}', \tilde{\sigma}_r)$ such that $\tilde{m}$ is not queried to neither the partial signing oracle nor the signing oracle. Thus $\mathcal{B}$ can simply break the unforgeability of $\mathcal{S}$ by outputting $(\tilde{m}, \tilde{\sigma}')$. □

8. Conclusion

Defining proper security models capturing possible realistic powers of an adversary is of significant importance in the study of optimistic fair exchange. In this paper we identified that the existing models failed to capture the reality that an adversary can have access to full signatures generated by the signer. We made an observation that the previous perception that a signing oracle can be simulated by the partial signing oracle and resolution oracle is indeed insufficient. That is to say, existing models are incomplete as they deprived dishonest users of the chance of observing the challenge signer's actual signatures. To make existing models more practical and complete, we proposed an enhanced model for optimistic fair exchange that explicitly provides the adversary with the signing oracle. Separations between existing chosen-key model and our model were demonstrated. Finally, we revisited two well-known approaches for constructing optimistic fair exchange schemes and showed that fortunately the two methodologies remain secure in our enhanced model.

[1] N. Asokan, M. Schunter, M. Waidner, Optimistic protocols for fair exchange, in: Proc. 4th ACM Conference on Computer and Communications Security, pp. 8–17.

[2] X. Huang, Y. Mu, W. Susilo, W. Wu, J. Zhou, R. H. Deng, Preserving transparency and accountability in optimistic fair exchange of digital signatures, IEEE Transactions on Information Forensics and Security 6 (2011) 498–512.

[3] N. Asokan, V. Shoup, M. Waidner, Optimistic fair exchange of digital signatures, Advances in Cryptology-Eurocrypt 1998, Lecture Notes in Computer Science 1403 (1998) 591 – 606.

[4] J. Camenisch, I. Damgård, Verifiable encryption, group encryption, and their applications to separable group signatures and signature sharing schemes, in: Proceedings of the 6th International Conference on the Theory and Application of Cryptology and Information Security: Advances in Cryptology, ASIACRYPT '00, Springer-Verlag, London, UK, 2000, pp. 331–345.

[5] D. Boneh, C. Gentry, B. Lynn, H. Shacham, Aggregate and verifiably encrypted signatures from bilinear maps, Advances in cryptology - Eurocrypt 2003, Lecture Notes in Computer Science 2656 (2003) 416–432.

[6] S. Lu, R. Ostrovsky, A. Sahai, H. Shacham, B. Waters, Sequential aggregate signatures and multisignatures without random oracles, in: EUROCRYPT, 2006., Springer, 2006, pp. 465–485.

[7] J. Zhang, J. Mao, A novel verifiably encrypted signature scheme without random oracle, in: Proceedings of the 3rd international conference on Information security practice and experience, ISPEC'07, Springer-Verlag, Berlin, Heidelberg, 2007, pp. 65–78.

[8] M. Rückert, D. Schröder, Security of verifiably encrypted signatures and a construction without random oracles, in: Proceedings of the 3rd International Conference Palo Alto on Pairing-Based Cryptography, Pairing '09, Springer-Verlag, Berlin, Heidelberg, 2009, pp. 17–34.

[9] Y. Dodis, L. Reyzin, Breaking and repairing optimistic fair exchange from podc 2003, in: Proceedings of the 3rd ACM workshop on Digital rights management, DRM '03, ACM, New York, NY, USA, 2003, pp. 47–54.

[10] Y. Dodis, P. J. Lee, D. H. Yum, Optimistic fair exchange in a multi-user setting, in: Proceedings of the 10th international conference on Practice

and theory in public-key cryptography, PKC'07, Springer-Verlag, Berlin, Heidelberg, 2007, pp. 118–133.

[11] Q. Huang, G. Yang, D. S. Wong, W. Susilo, Efficient optimistic fair exchange secure in the multi-user setting and chosen-key model without random oracles, in: Proceedings of the 2008 The Cryptopgraphers' Track at the RSA conference on Topics in cryptology, CT-RSA'08, Springer-Verlag, Berlin, Heidelberg, 2008, pp. 106–120.

[12] H. Zhu, W. Susilo, Y. Mu, Multi-party stand-alone and setup-free verifiably committed signatures, in: [29], pp. 134–149.

[13] B. Barak, R. Canetti, J. B. Nielsen, R. Pass, Universally composable protocols with relaxed set-up assumptions., in: FOCS'04, pp. 186–195.

[14] D. Boneh, M. K. Franklin, Identity-based encryption from the weil pairing, in: J. Kilian (Ed.), CRYPTO, volume 2139 of *Lecture Notes in Computer Science*, Springer, 2001, pp. 213–229.

[15] D. Boneh, X. Boyen, Short signatures without random oracles and the sdh assumption in bilinear groups, J. Cryptology 21 (2008) 149–177.

[16] X. Boyen, B. Waters, Full-domain subgroup hiding and constant-size group signatures, in: [29], pp. 1–15.

[17] Q. Huang, D. S. Wong, Short convertible undeniable signature in the standard model, in: F. Bao, J. Weng (Eds.), ISPEC, volume 6672 of *Lecture Notes in Computer Science*, Springer, 2011, pp. 257–272.

[18] D. Hofheinz, E. Kiltz, Programmable hash functions and their applications, J. Cryptology 25 (2012) 484–527.

[19] Q. Huang, D. S. Wong, W. Susilo, Efficient designated confirmer signature and dcs-based ambiguous optimistic fair exchange, IEEE Transactions on Information Forensics and Security 6 (2011) 1233–1247.

[20] D. Chaum, Designated confirmer signatures, in: A. D. Santis (Ed.), EUROCRYPT, volume 950 of *Lecture Notes in Computer Science*, Springer, 1994, pp. 86–91.

[21] M. Bellare, G. Neven, Multi-signatures in the plain public-key model and a general forking lemma, in: Proceedings of the 13th ACM conference on Computer and communications security, CCS '06, ACM, New York, NY, USA, 2006, pp. 390–399.

[22] D. Boneh, B. Lynn, H. Shacham, Short signatures from the weil pairing, J. Cryptology 17 (2004) 297–319.

[23] A. Bender, J. Katz, R. Morselli, Ring signatures: Stronger definitions, and constructions without random oracles, in: S. Halevi, T. Rabin (Eds.), TCC, volume 3876 of *Lecture Notes in Computer Science*, Springer, 2006, pp. 60–79.

[24] B. Waters, Efficient identity-based encryption without random oracles, in: R. Cramer (Ed.), EUROCRYPT, volume 3494 of *Lecture Notes in Computer Science*, Springer, 2005, pp. 114–127.

[25] A. Sahai, Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security, in: Proceedings of the 40th Annual Symposium on Foundations of Computer Science, FOCS '99, IEEE Computer Society, Washington, DC, USA, 1999, pp. 543–553.

[26] S. Goldwasser, S. Micali, R. L. Rivest, A digital signature scheme secure against adaptive chosen-message attacks, SIAM J. Comput. 17 (1988) 281–308.

[27] C. Rackoff, D. R. Simon, Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack, in: Proceedings of the 11th Annual International Cryptology Conference on Advances in Cryptology, CRYPTO '91, Springer-Verlag, London, UK, UK, 1992, pp. 433–444.

[28] X. Boyen, Mesh signatures, in: M. Naor (Ed.), EUROCRYPT, volume 4515 of *Lecture Notes in Computer Science*, Springer, 2007, pp. 210–227.

[29] T. Okamoto, X. Wang (Eds.), Public Key Cryptography - PKC 2007, 10th International Conference on Practice and Theory in Public-Key Cryptography, Beijing, China, April 16-20, 2007, Proceedings, volume 4450 of *Lecture Notes in Computer Science*, Springer, 2007.